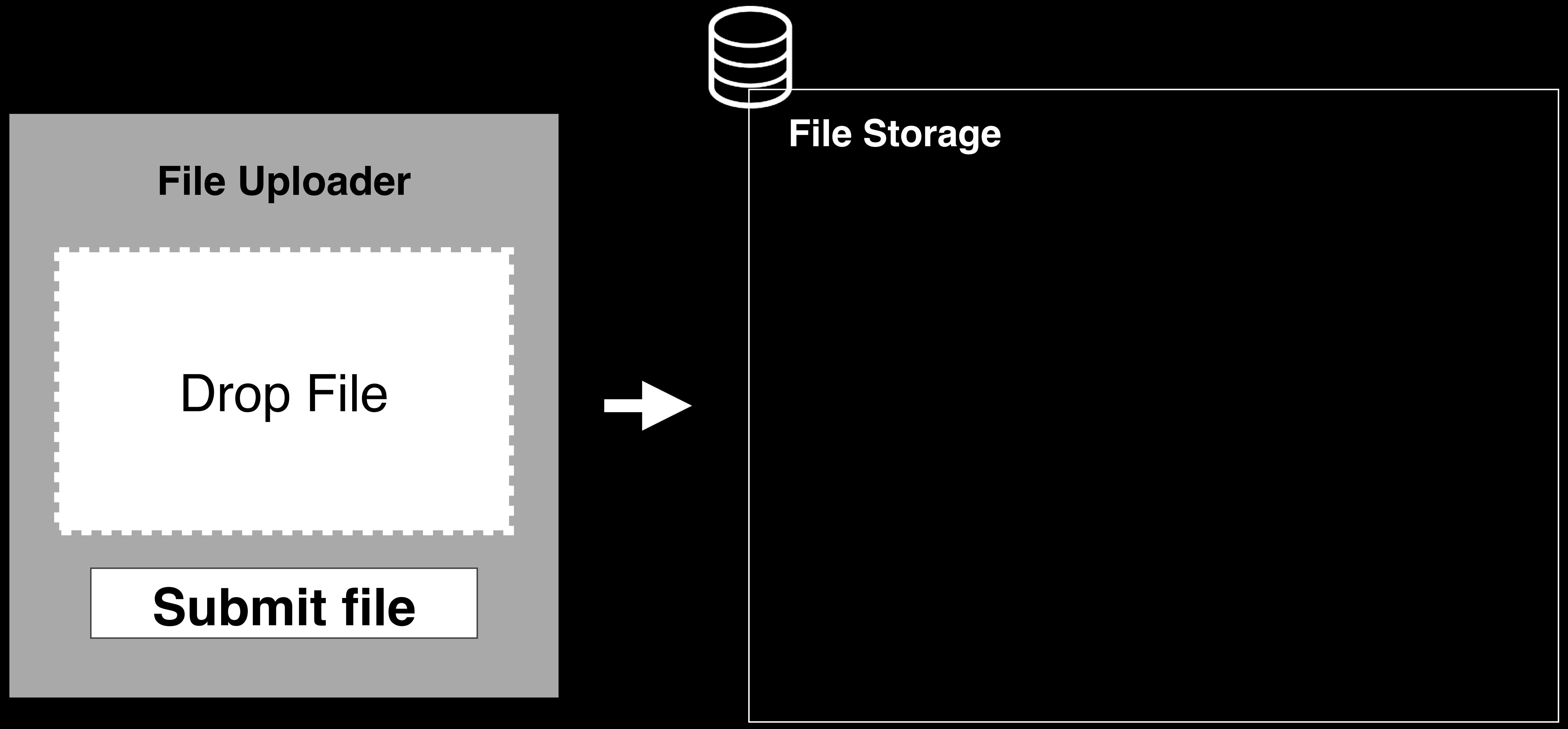
"filename" in Content-Disposition is a landmine vulnerability caused by ambiguous requirements

CODE BLUE 2023

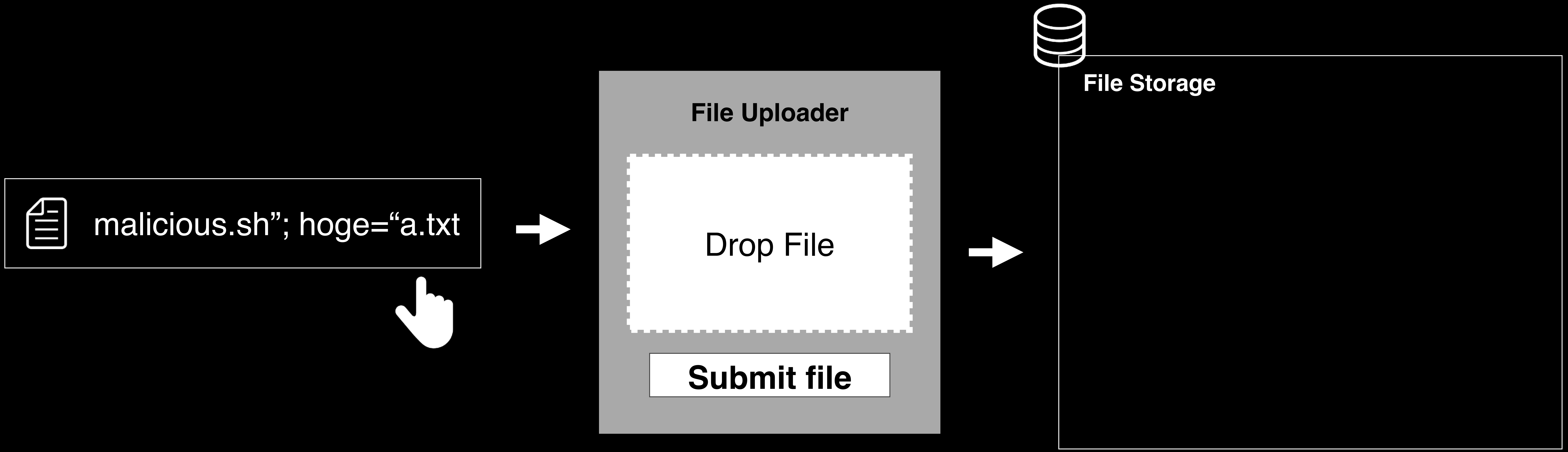
Who am I?

- Motoyasu Saburi
- Security Engineer @ PayPay
    - Blue Team
    - Anti Phishing Scam
    - SOC

Upload File



Upload File



malicious.sh”; hoge=“a.txt

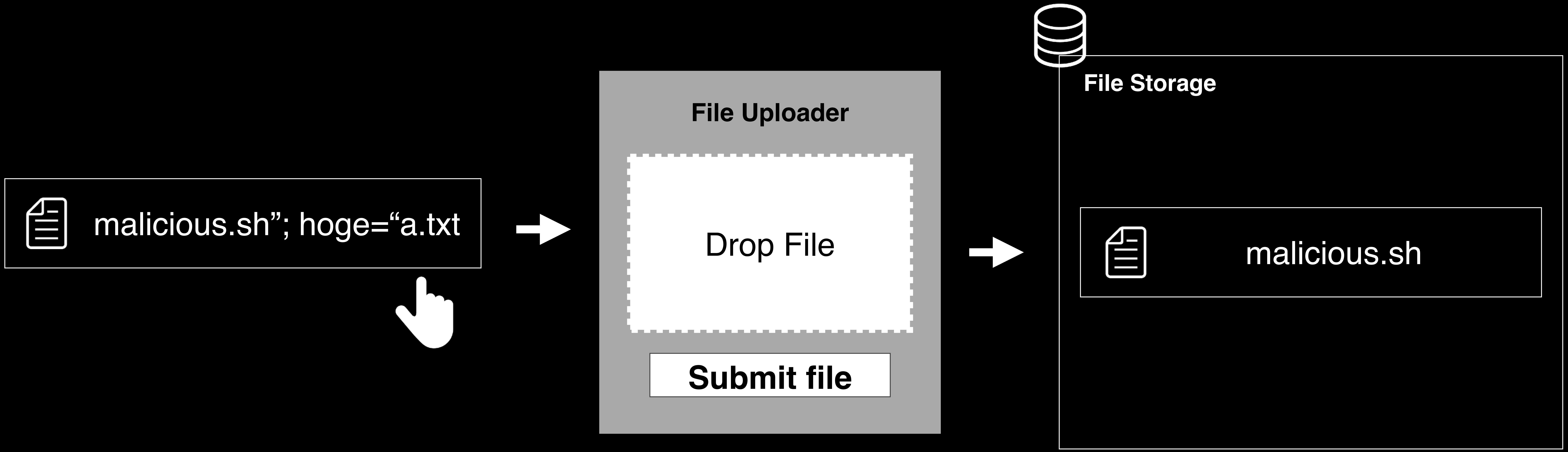
File Uploader

Drop File

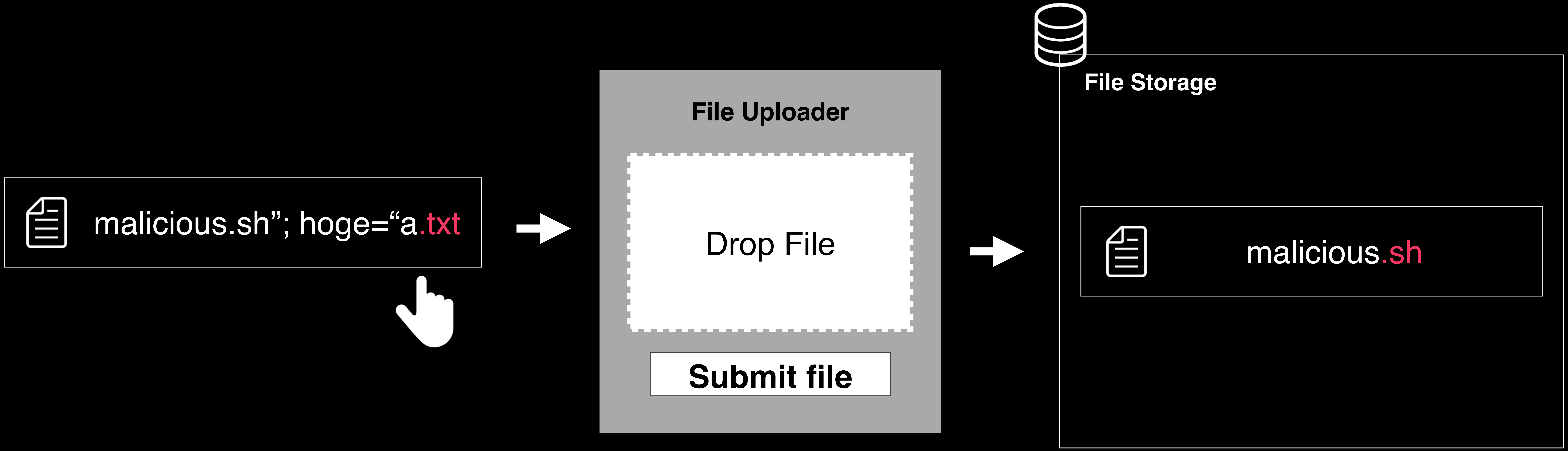
Submit file

File Storage

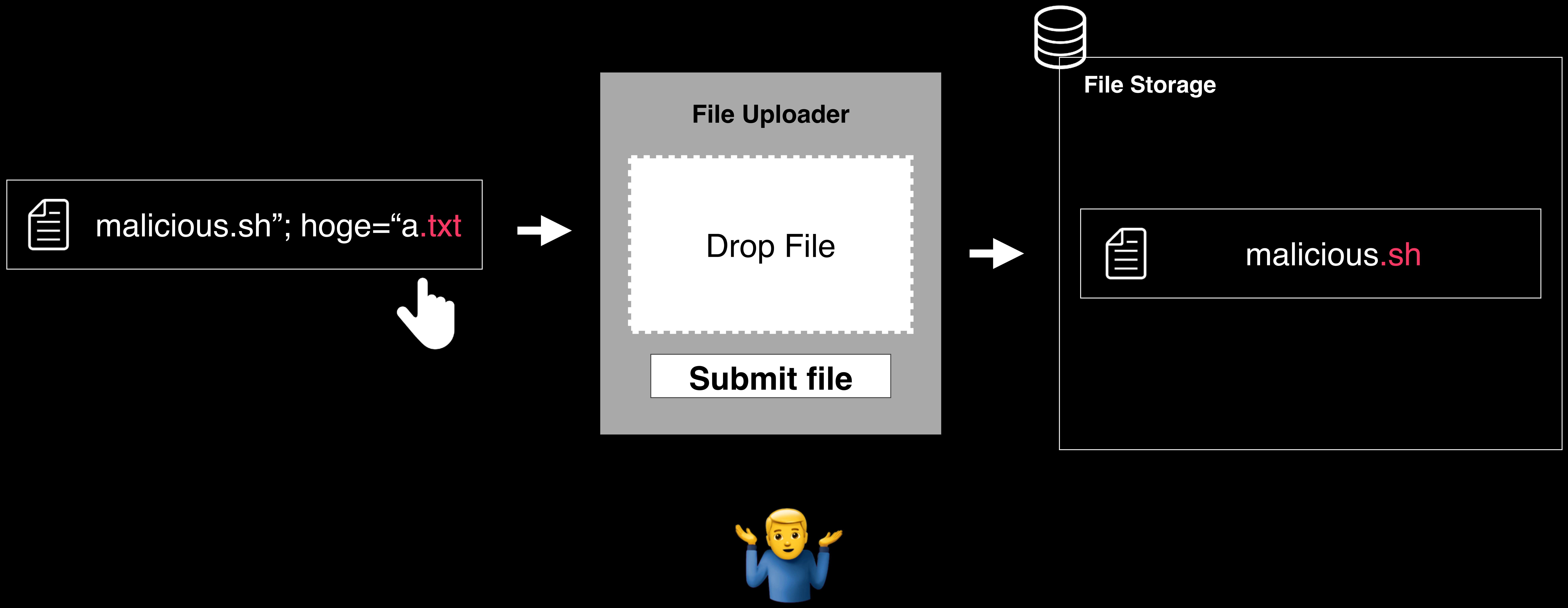
Upload File


malicious.sh”; hoge=“a.txt
File Uploader
Drop File
Submit file
File Storage
malicious.sh

Upload File


malicious.sh”; hoge=“a.txt
File Uploader
Drop File
Submit file
File Storage
malicious.sh

Upload File



malicious.sh”; hoge=“a.txt

File Uploader

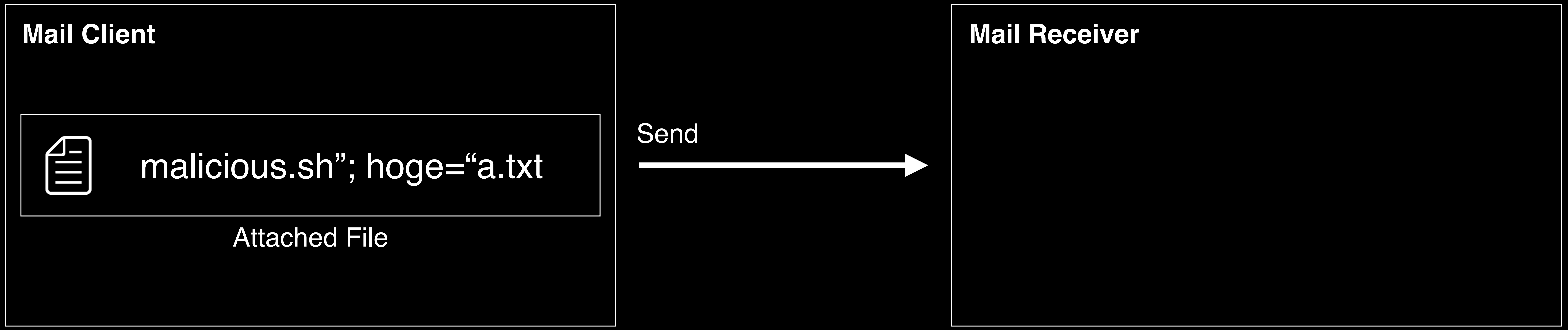
Drop File

Submit file

File Storage

malicious.sh

Send Mail



Mail Client

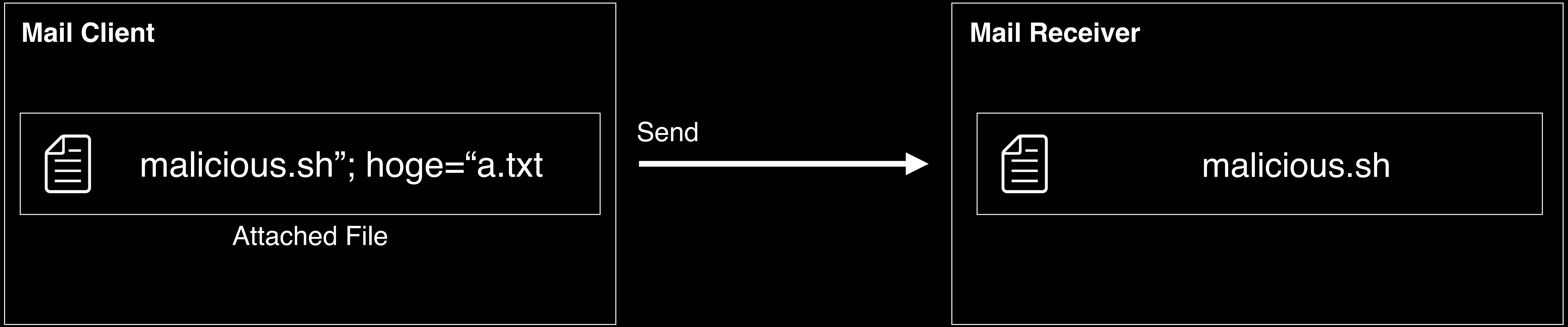
malicious.sh”; hoge=“a.txt

Attached File

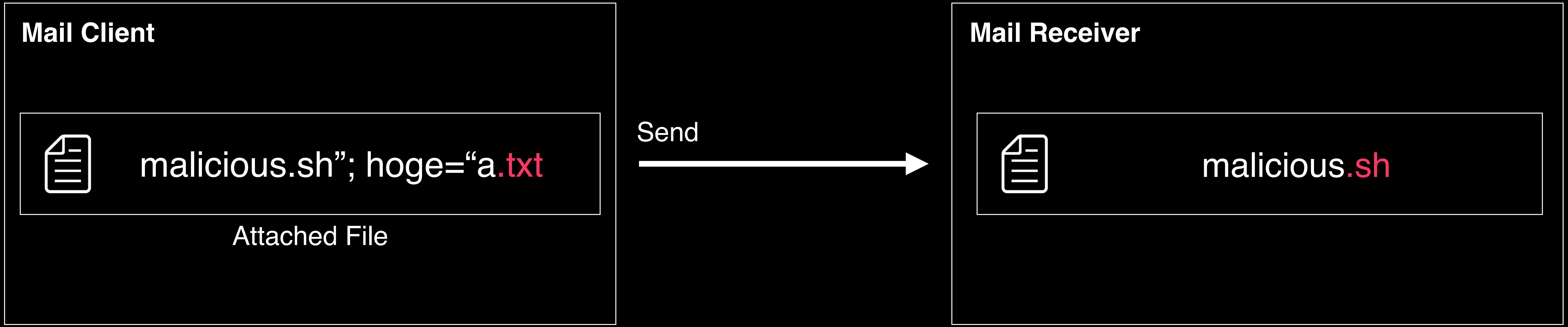
Send

Mail Receiver

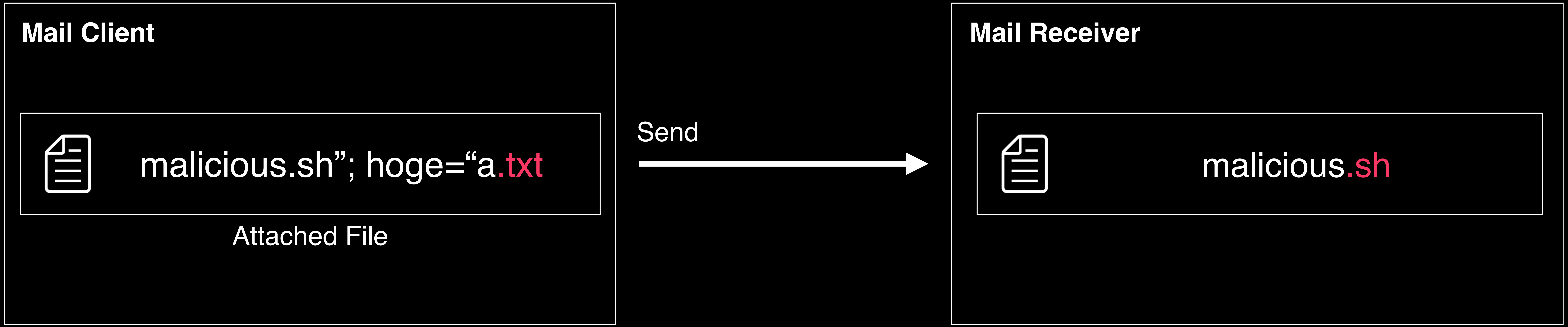
Send Mail


Mail Client
malicious.sh”; hoge=“a.txt
Attached File
Send
Mail Receiver
malicious.sh

Send Mail


Mail Client
malicious.sh"; hoge="a.txt
Attached File
Send
Mail Receiver
malicious.sh

Send Mail



Mail Client

malicious.sh”; hoge=“a.txt

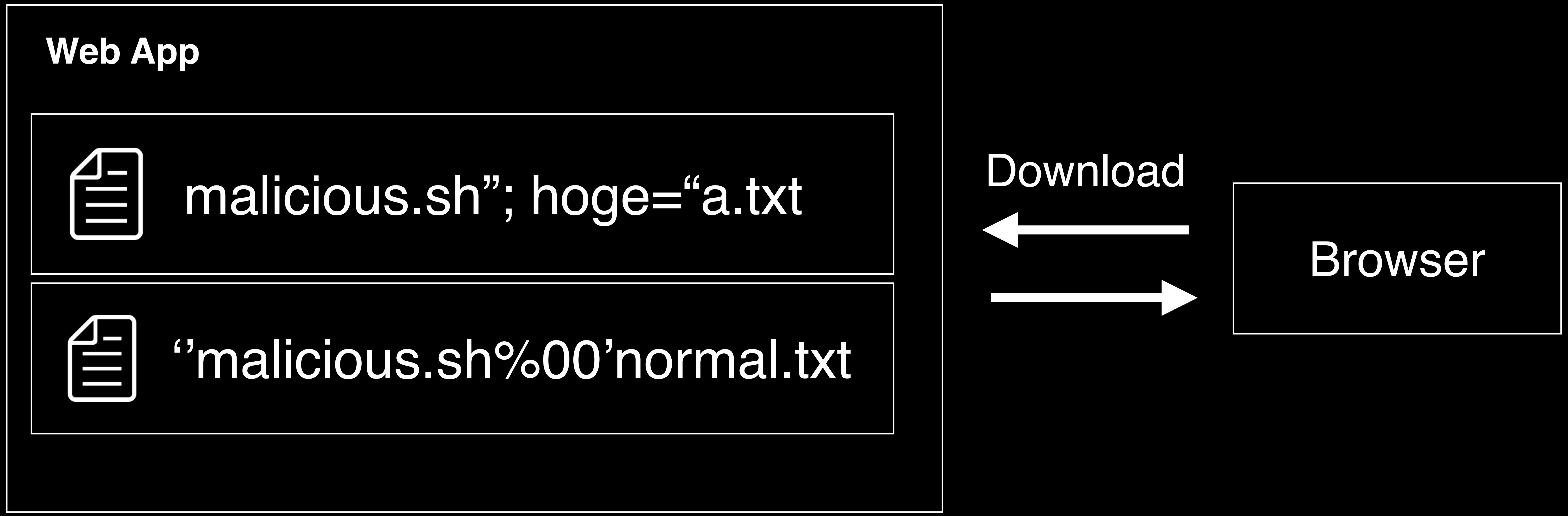
Attached File

Send

Mail Receiver

malicious.sh

Download File



Web App

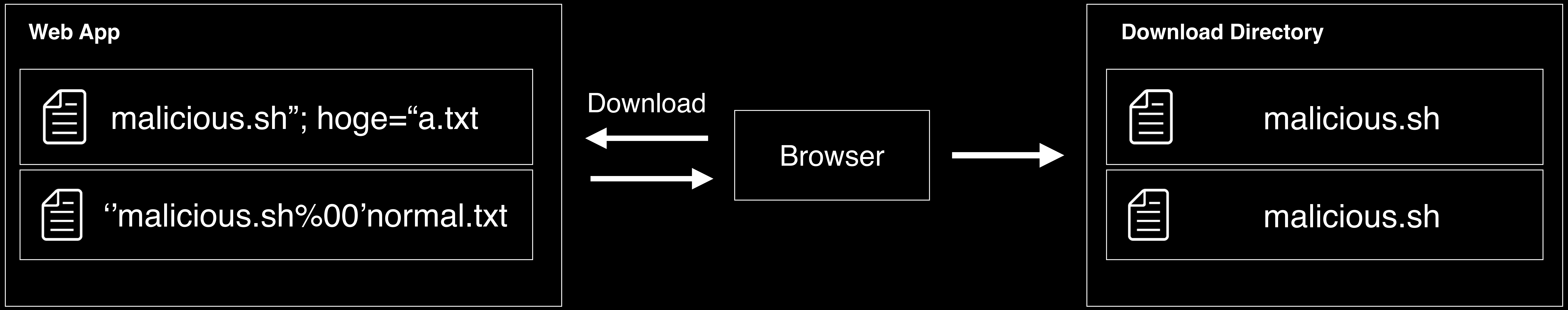
malicious.sh”; hoge=“a.txt

‘’malicious.sh%00’normal.txt

Download

Browser

Download File



Web App

malicious.sh”; hoge=“a.txt

‘’malicious.sh%00’normal.txt

Download

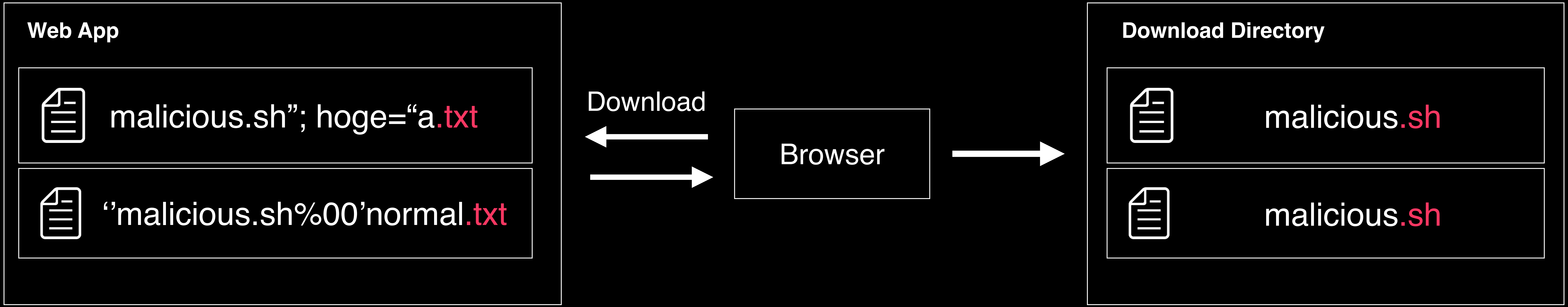
Browser

Download Directory

malicious.sh

malicious.sh

Download File



Web App

malicious.sh”; hoge=“a.txt

‘’malicious.sh%00’normal.txt

Download

Browser

Download Directory

malicious.sh

malicious.sh

TL;DR

- ***Content-Dispositon*** have an incomprehensible escaping requirement.
  - **RFC**: “**Unclear**” requirements.
  - **WhatWG HTML Spec**: “**Clear**” requirements.

- **I discovered many vulnerabilities.**
  - 1 Browser
  - 2 Programming Lang
  - 6 Web Framework
  - 5 HTTP Client
  - 2 Mailer Library

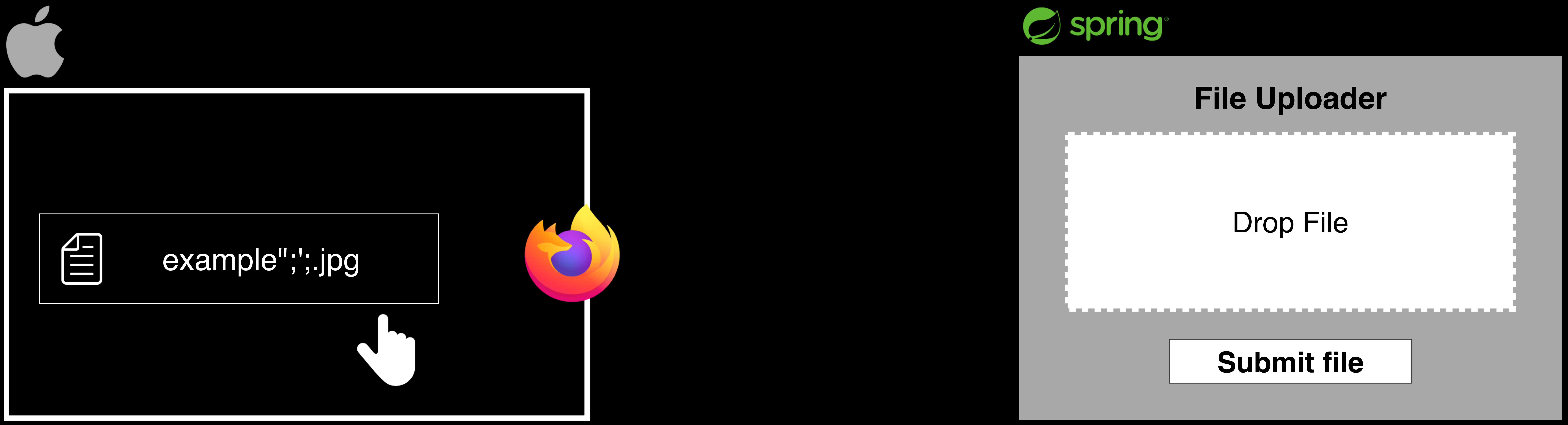
- **Content-Disposition “filename” requires escaping 3 char**
  - \r —> %0d or \\r
  - \n —> %0a or \\a
  - “ —> %22 or \”

Signs of a problem

- I was security-testing the Web Service (Spring).

Signs of a problem

- I was security-testing the Web Service (Spring).



example";';.jpg

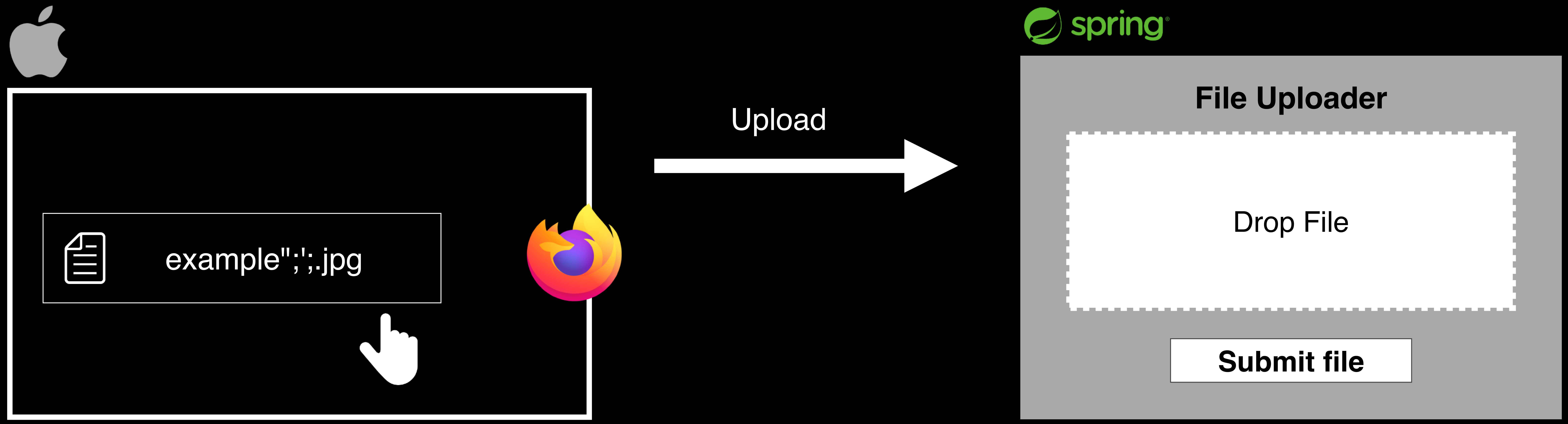
File Uploader

Drop File

Submit file

Signs of a problem

- I was security-testing the Web Service (Spring).



example";';.jpg

Upload

spring

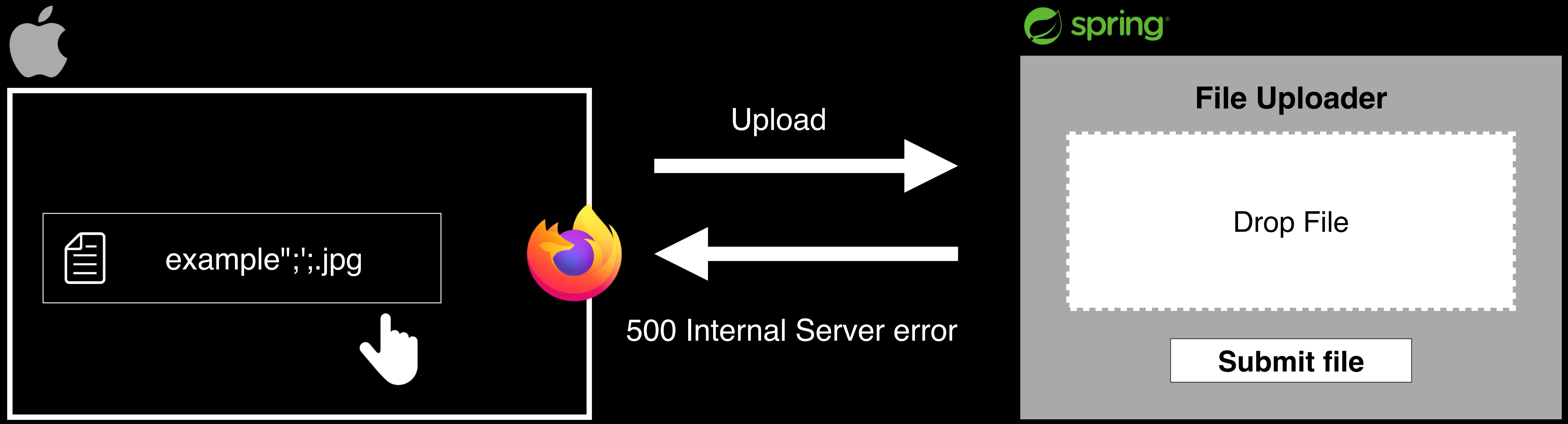
File Uploader

Drop File

Submit file

Signs of a problem

- I was security-testing the Web Service (Spring).



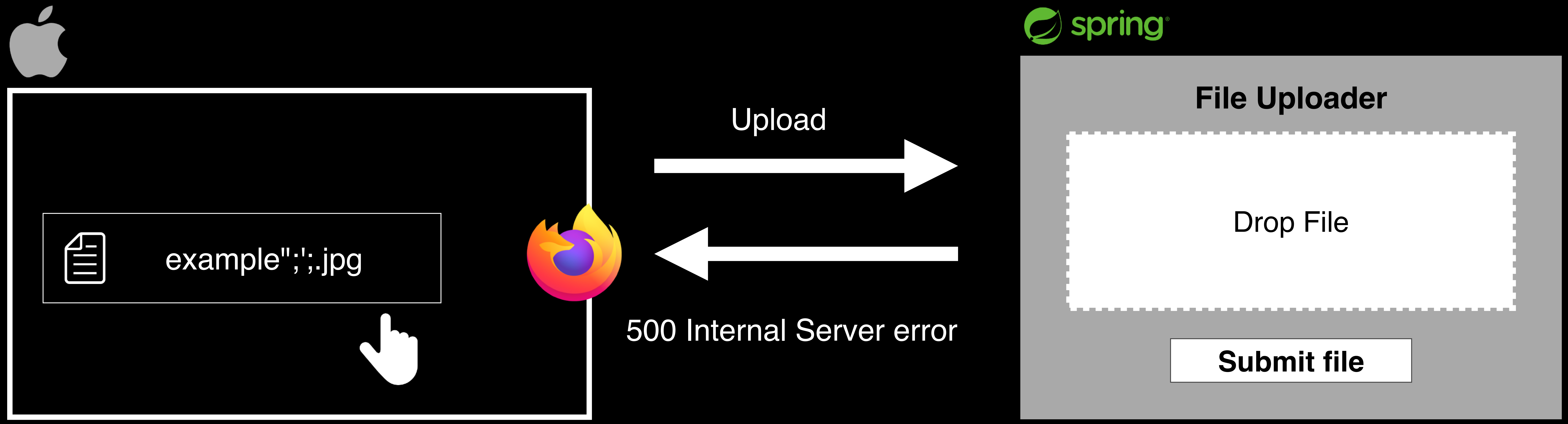
example";';.jpg

Upload

500 Internal Server error

Signs of a problem

- I was security-testing the Web Service (Spring).



example";';.jpg

Upload

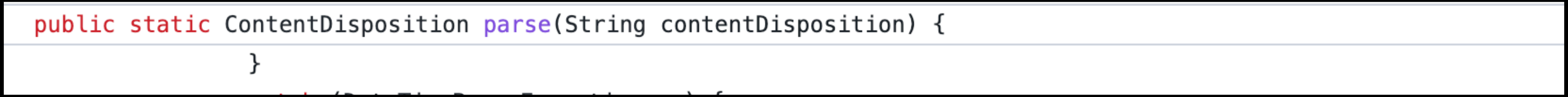
500 Internal Server error

I tried various cases, but any Injections failed.

Read Server Error

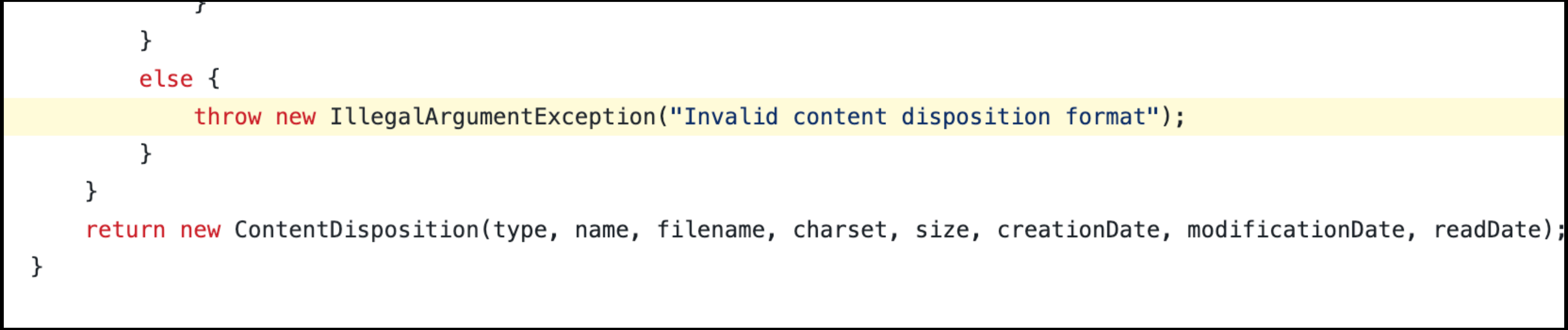
Read Server Error

https://github.com/spring-projects/spring-framework/blob/4f05da7fed7e55d0744a91e4ac384d8f5df6e665/spring-web/src/main/java/org/springframework/http/ContentDisposition.java#L316



```
public static ContentDisposition parse(String contentDisposition) {
                }
```

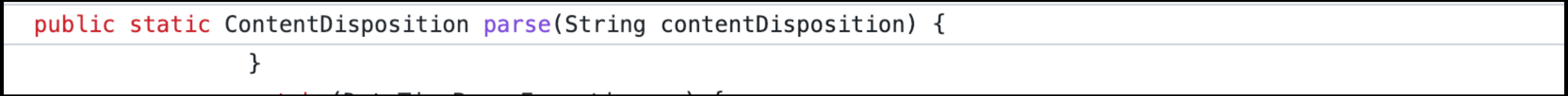
…



```
            }
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

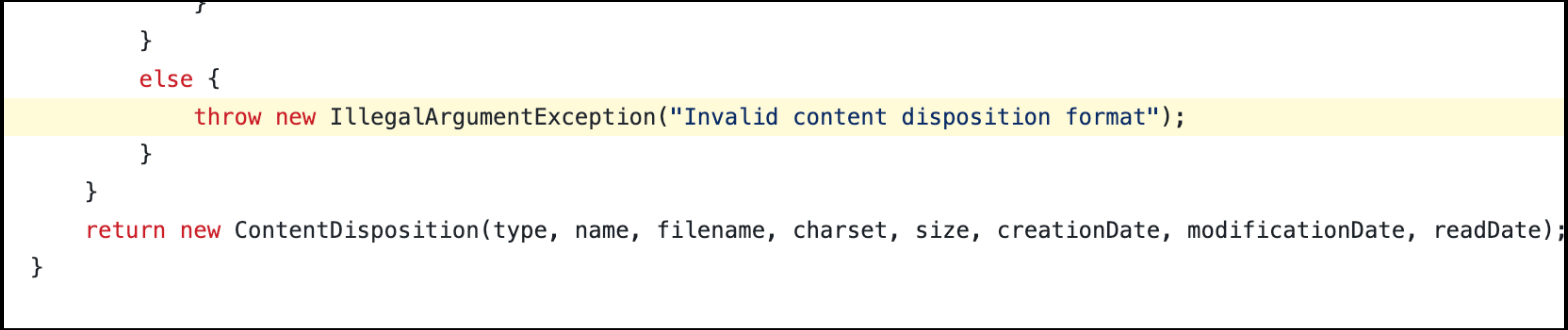
Read Server Error

https://github.com/spring-projects/spring-framework/blob/4f05da7fed7e55d0744a91e4ac384d8f5df6e665/spring-web/src/main/java/org/springframework/http/ContentDisposition.java#L316



```
public static ContentDisposition parse(String contentDisposition) {
                    }
```

…



```
                }
            }
            else {
                throw new IllegalArgumentException("Invalid content disposition format");
            }
        }
        return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
    }
```

What is **Content-Disposition**?

Content-Disposition

Content-Disposition - HTTP | MDN

> In a regular HTTP response, the Content-Disposition response header is a header indicating if the content is expected to be displayed inline in the browser, that is, as a Web page or as part of a Web page, or as an attachment, that is downloaded and saved locally.
…
> In a multipart/form-data body, the HTTP Content-Disposition general header is a header that must be used on each subpart of a multipart body to give information about the field it applies to.
…
> The Content-Disposition header is defined in the larger context of MIME messages for email, but only a subset of the possible parameters apply to HTTP forms and POST requests. Only the value form-data, as well as the optional directive name and filename, can be used in the HTTP context.

Content-Disposition

Content-Disposition - HTTP | MDN

> In a regular HTTP response, the Content-Disposition response header is a header indicating if the content is expected to be displayed inline in the browser, that is, as a Web page or as part of a Web page, or as an attachment, that is downloaded and saved locally.

Control whether the content is downloaded or displayed as-is in the browser.

…

> In a multipart/form-data body, the HTTP Content-Disposition general header is a header that must be used on each subpart of a multipart body to give information about the field it applies to.

Defines meta-information for fields in multipart/form-data requests

…

> The Content-Disposition header is defined in the larger context of MIME messages for email, but only a subset of the possible parameters apply to HTTP forms and POST requests. Only the value form-data, as well as the optional directive name and filename, can be used in the HTTP context.

Defines meta-information for fields in multipart requests on email

Content-Disposition

Content-Disposition - HTTP | MDN

> In a regular HTTP response, the Content-Disposition response header is a header indicating if the content is expected to be displayed inline in the browser, that is, as a Web page or as part of a Web page, or as an attachment, that is downloaded and saved locally.

Control whether the content should be displayed as-is in the browser.

HTTP Response Header

…

> In a multipart/form-data body, the HTTP Content-Disposition general header is a header that must be used on each subpart of a multipart body to give information about the field it applies to.

Defines meta-information for fields in multipart/form-data requests

…

> The Content-Disposition header is defined in the larger context of MIME messages for email, but only a subset of the possible parameters apply to HTTP forms and POST requests. Only the value form-data, as well as the optional directive name and filename, can be used in the HTTP context.

Defines meta-information for fields in multipart requests on email

Content-Disposition

[Content-Disposition - HTTP | MDN]()

Control whether the content is to be displayed as-is in the browser.

HTTP Response Header

…

Defines meta-information for fields in multipart/form-data requests

…

Header in multipart > part

Defines meta-information for fields in multipart requests on email

Content-Disposition

Content-Disposition - HTTP | MDN

Control whether the content is to be displayed as-is in the browser.

HTTP Response Header

…

Defines meta-information for fields in multipart/form-data requests

…

Header in multipart > part

Defines meta-information for fields in multipart requests on email

Content-Disposition - multipart

Multipart is the format used in HTTP Request and Email.

Content-Disposition - multipart

Multipart is the format used in HTTP Request and Email.

Header

```
Content-Type: multipart/form-data; boundary=--foobarRandom1234
```

Body

```
–foobarRandom1234
Content-Disposition: form-data; name=“user_name”
Content-Type: text/plain

FooBar
--foobarRandom1234
Content-Disposition: form-data; name=“email”; filename=“email.txt”

bob@example.com
--foobarRandom1234
Content-Disposition: form-data; name=“desc”; filename=“hello.txt”

This is file data.
–foobarRandom1234--
```

Content-Disposition - multipart

Multipart is the format used in HTTP Request and Email.

Header

```
Content-Type: multipart/form-data; boundary=--foobarRandom1234
```

Body

```
–foobarRandom1234
Content-Disposition: form-data; name=“user_name”
Content-Type: text/plain

FooBar
--foobarRandom1234
Content-Disposition: form-data; name=“email”; filename=“email.txt”

bob@example.com
--foobarRandom1234
Content-Disposition: form-data; name=“desc”; filename=“hello.txt”

This is file data.
–foobarRandom1234--
```

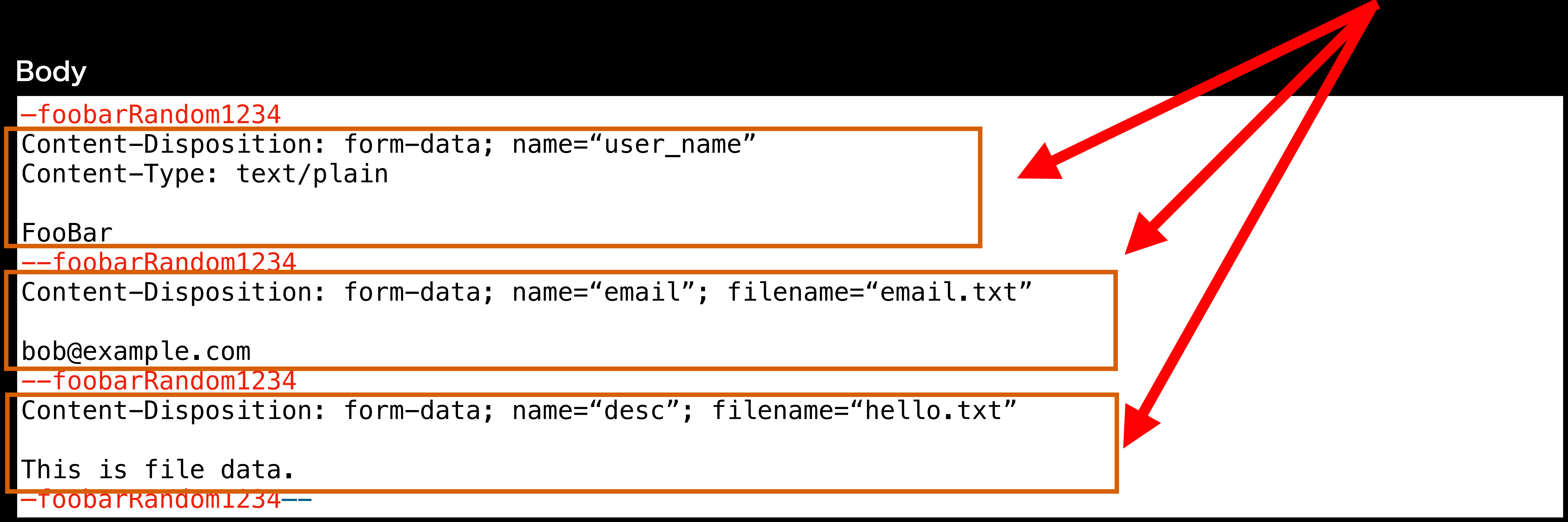
Content-Disposition - multipart

Multipart is the format used in HTTP Request and Email.

Header

```
Content-Type: multipart/form-data; boundary=--foobarRandom1234
```

Body



```
–foobarRandom1234
Content-Disposition: form-data; name=“user_name”
Content-Type: text/plain

FooBar
--foobarRandom1234
Content-Disposition: form-data; name=“email”; filename=“email.txt”

bob@example.com
--foobarRandom1234
Content-Disposition: form-data; name=“desc”; filename=“hello.txt”

This is file data.
–foobarRandom1234--
```

Content-Disposition - multipart

Multipart is the format used in HTTP Request and Email.

Header

```
Content-Type: multipart/form-data; boundary=--foobarRandom1234
```

Body

```
–foobarRandom1234
Content-Disposition: form-data; name=“user_name”
Content-Type: text/plain

FooBar
--foobarRandom1234
Content-Disposition: form-data; name=“email”; filename=“email.txt”

bob@example.com
--foobarRandom1234
Content-Disposition: form-data; name=“desc”; filename=“hello.txt”

This is file data.
–foobarRandom1234--
```

Content-Disposition - multipart

Multipart is the format used in HTTP Request and Email.

Header

```
Content-Type: multipart/form-data; boundary=--foobarRandom1234
```

Body

```
–foobarRandom1234
Content-Disposition: form-data; name=“user_name”
Content-Type: text/plain

FooBar
--foobarRandom1234
Content-Disposition: form-data; name=“email”; filename=“email.txt”

bob@example.com
--foobarRandom1234
Content-Disposition: form-data; name=“desc”; filename=“hello.txt”

This is file data.
–foobarRandom1234--
```

Content-Disposition - multipart

Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```

Content-Disposition - multipart

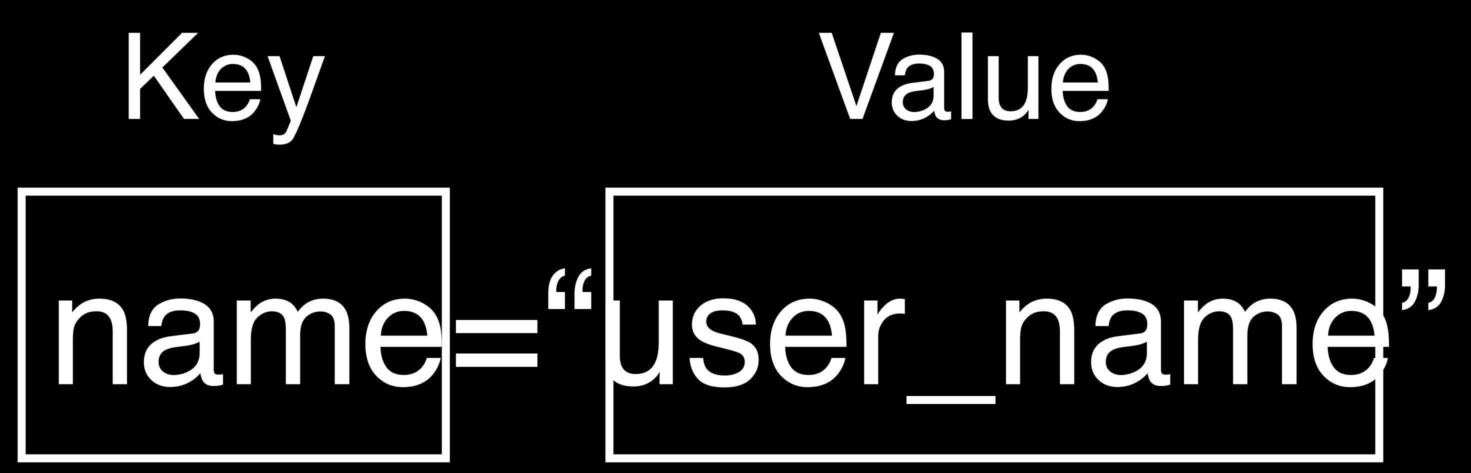
```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```

Disposition-Type:

- **form-data** : Use with form-data
- **inline** : Display file in browser, etc.
- **attachment** : Downloading files
- etc

Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```



Key Value

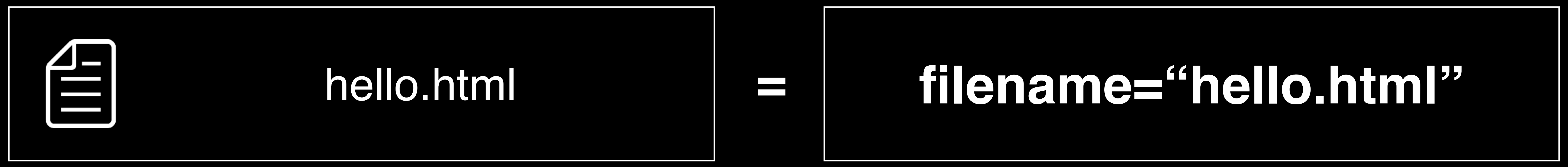
name=“user_name”

Content-Disposition fields are basically described in **key="Value"** format.

"name" indicates the name of the field in the form.

Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```

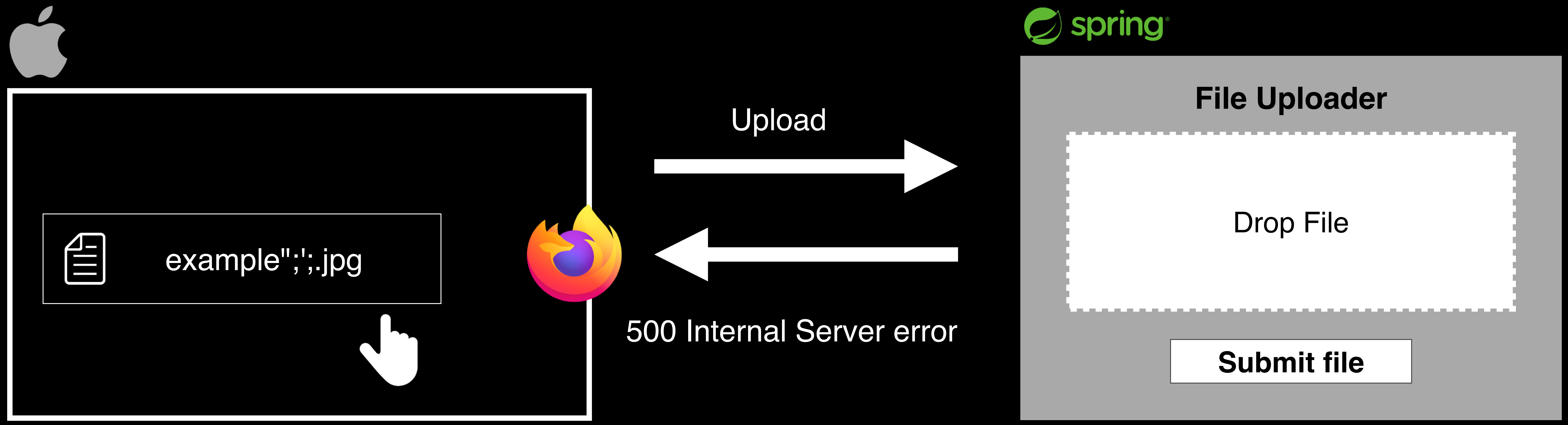


hello.html = filename=“hello.html”

filename=""; is used to specify a file name when including a file in a form.

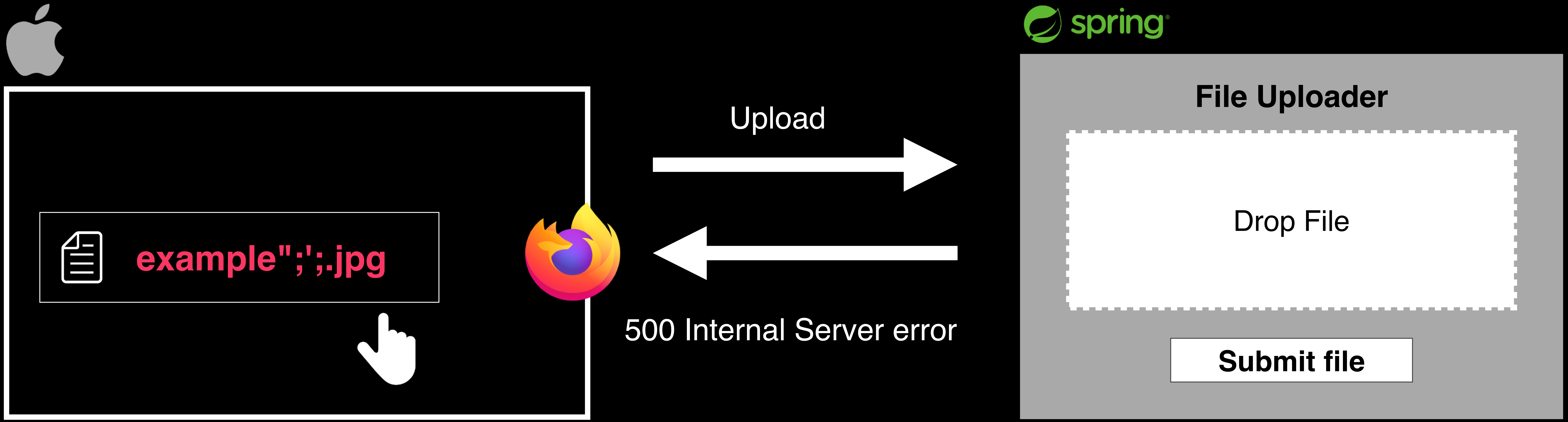
Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```



Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```



example";';.jpg

Upload

500 Internal Server error

spring

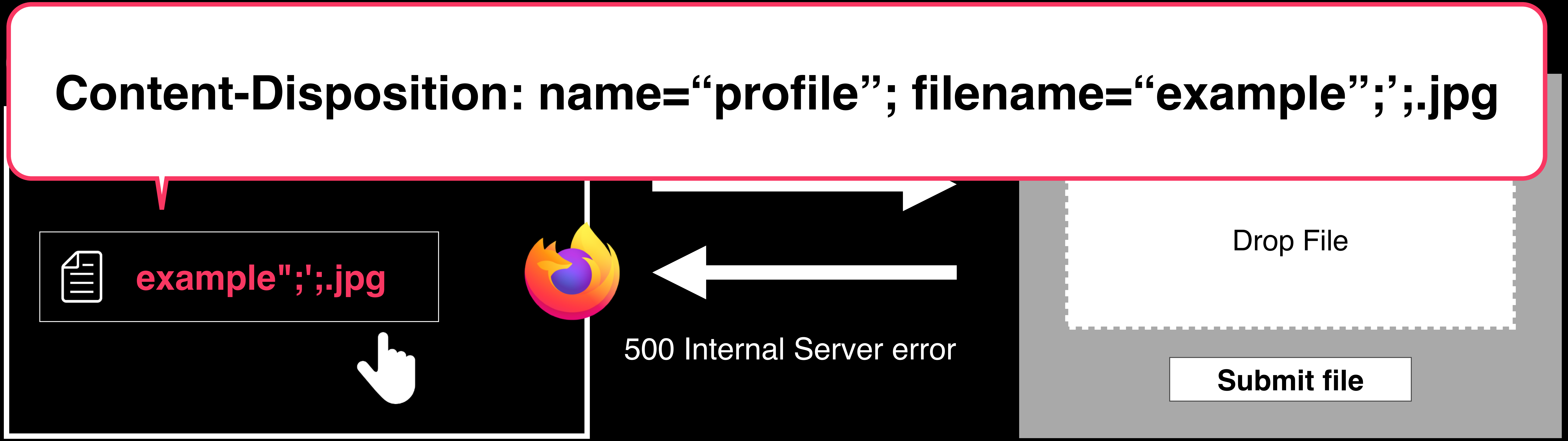
File Uploader

Drop File

Submit file

Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```

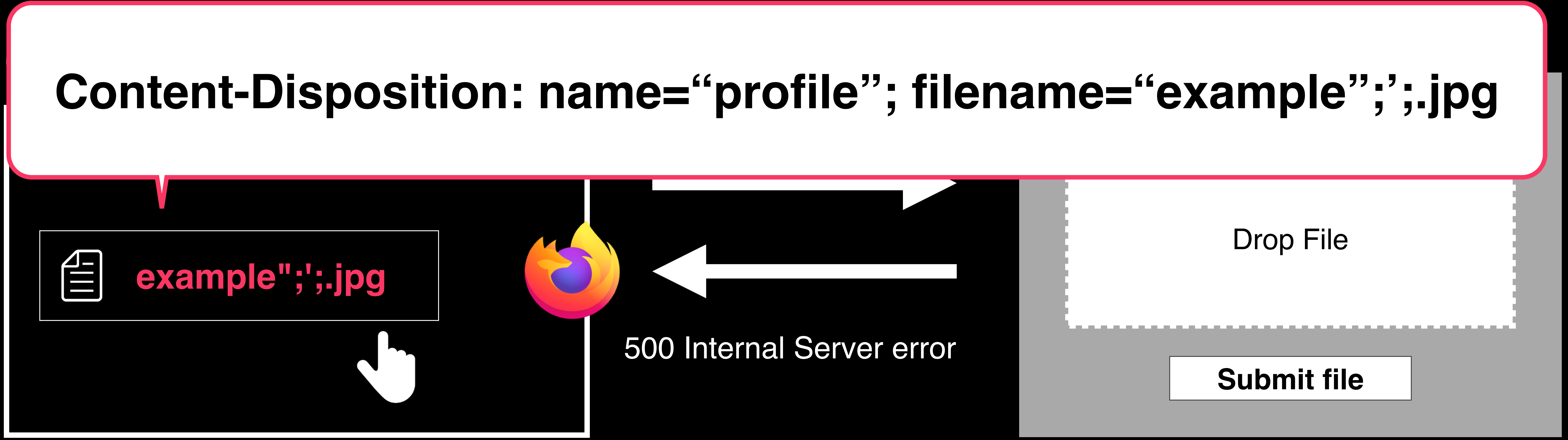


Content-Disposition: name=“profile”; filename=“example”;’;.jpg

Content-Disposition - multipart

```
Content-Disposition: form-data; name=“XXX”; filename=“XXX.txt”;
```

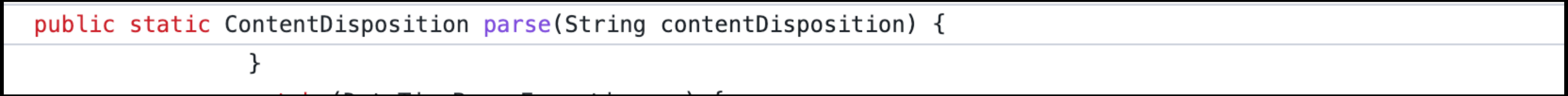
Why did the error occur?



Content-Disposition: name=“profile”; filename=“example”;’;.jpg

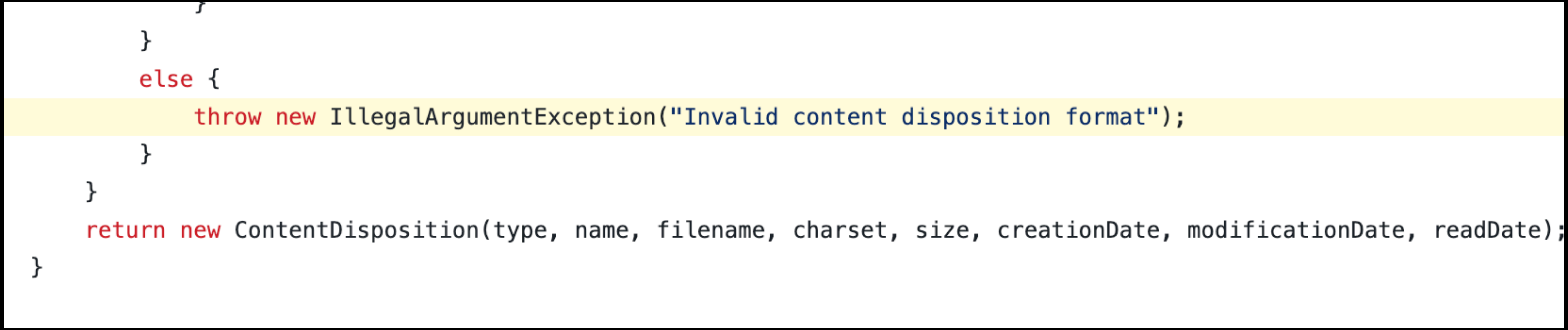
Read Server Error

https://github.com/spring-projects/spring-framework/blob/4f05da7fed7e55d0744a91e4ac384d8f5df6e665/spring-web/src/main/java/org/springframework/http/ContentDisposition.java#L316



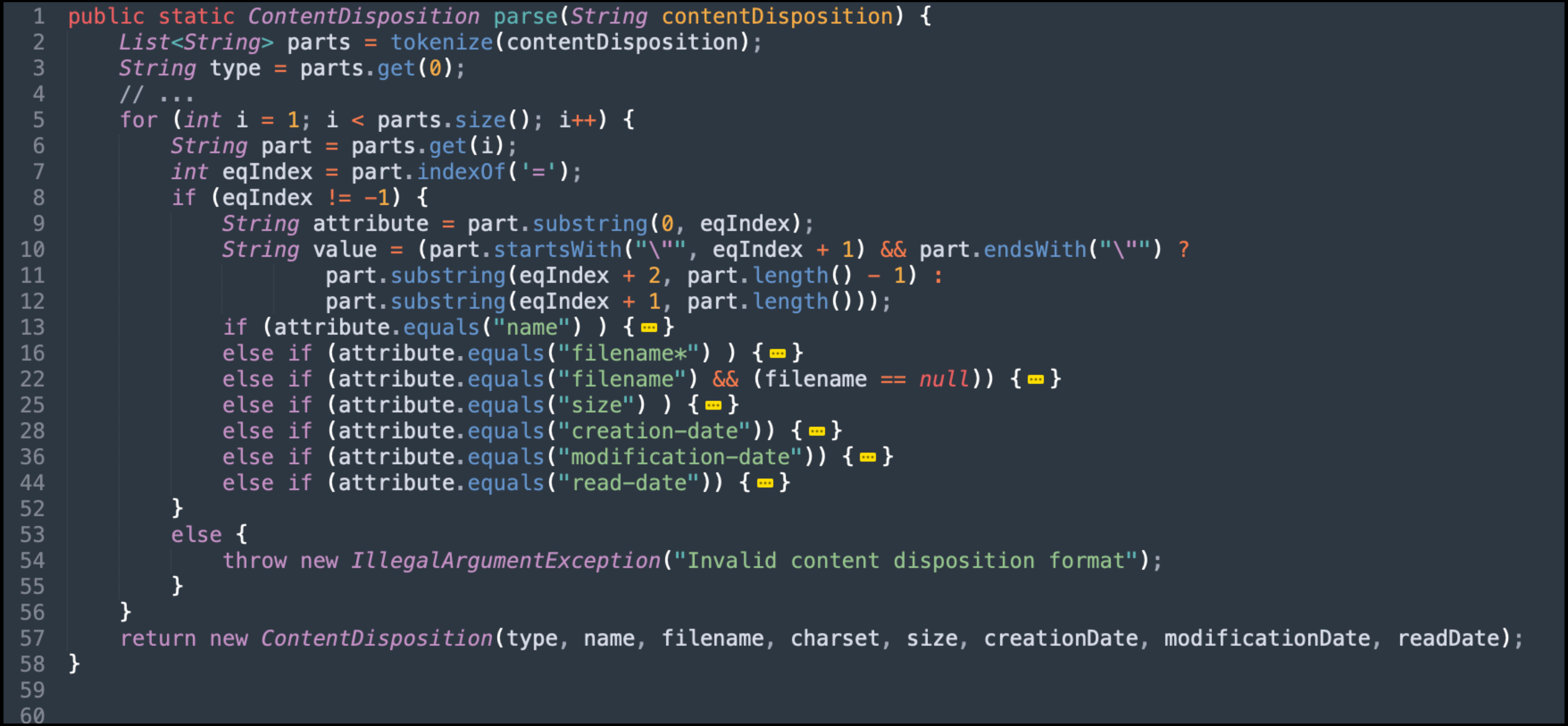
```java
public static ContentDisposition parse(String contentDisposition) {
                }
```

…



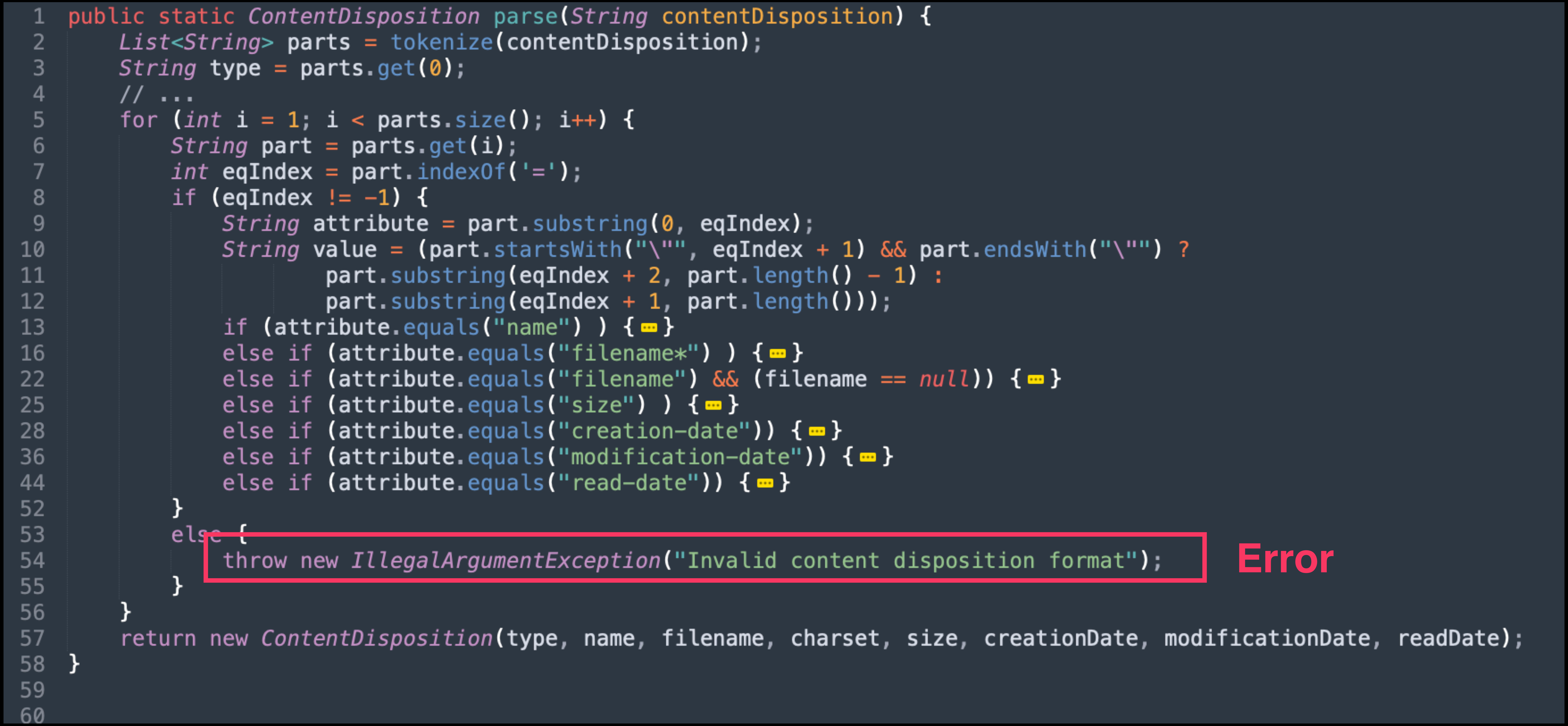
```java
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

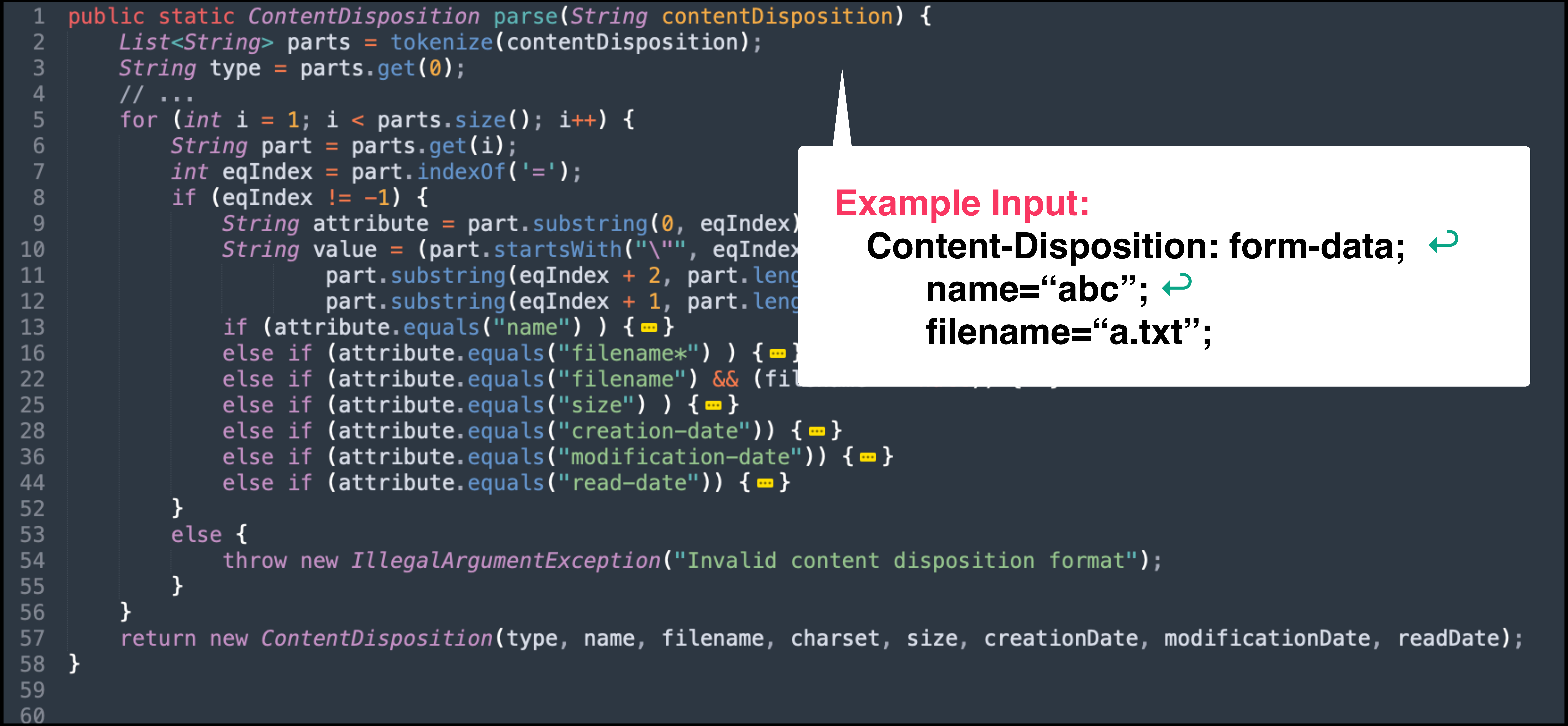
Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Error

Read Server Error - part 2

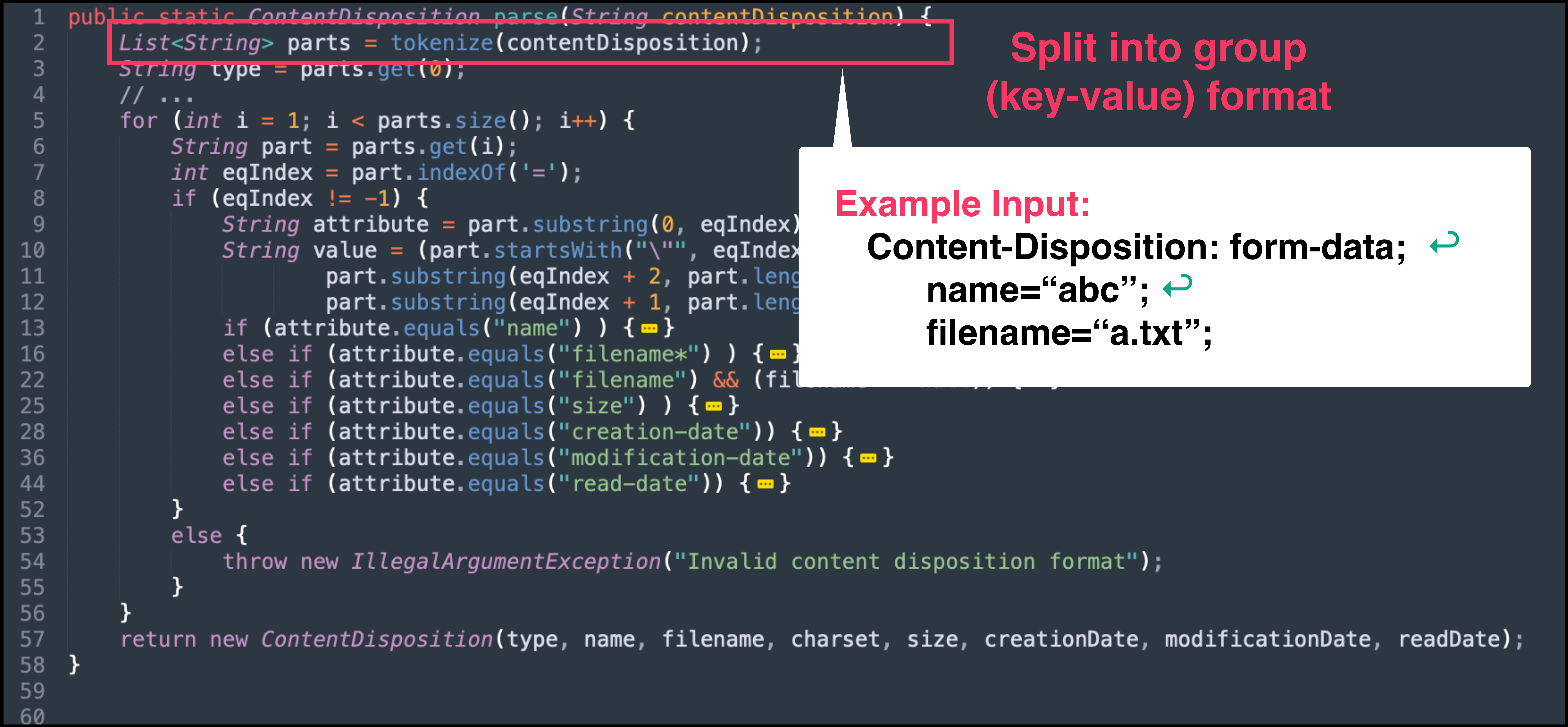


```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex)
            String value = (part.startsWith("\"", eqIndex
                    part.substring(eqIndex + 2, part.leng
                    part.substring(eqIndex + 1, part.leng
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (fil
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Example Input:

Content-Disposition: form-data; ↩
name=“abc”; ↩
filename=“a.txt”;

Read Server Error - part 2



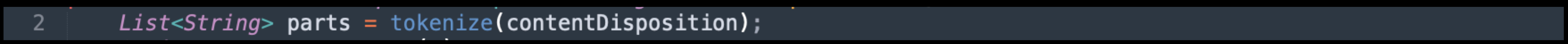
```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex)
            String value = (part.startsWith("\"", eqIndex
                    part.substring(eqIndex + 2, part.leng
                    part.substring(eqIndex + 1, part.leng
            if (attribute.equals("name") ) {…}
            else if (attribute.equals("filename*") ) {…}
            else if (attribute.equals("filename") && (fil
            else if (attribute.equals("size") ) {…}
            else if (attribute.equals("creation-date")) {…}
            else if (attribute.equals("modification-date")) {…}
            else if (attribute.equals("read-date")) {…}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Split into group (key-value) format

Example Input:

Content-Disposition: form-data; ↩
name="abc"; ↩
filename="a.txt";

Read Server Error - part 2

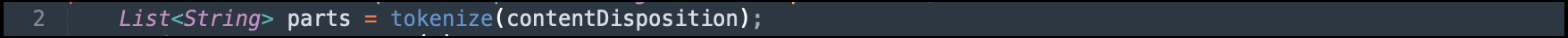


```
    List<String> parts = tokenize(contentDisposition);
```

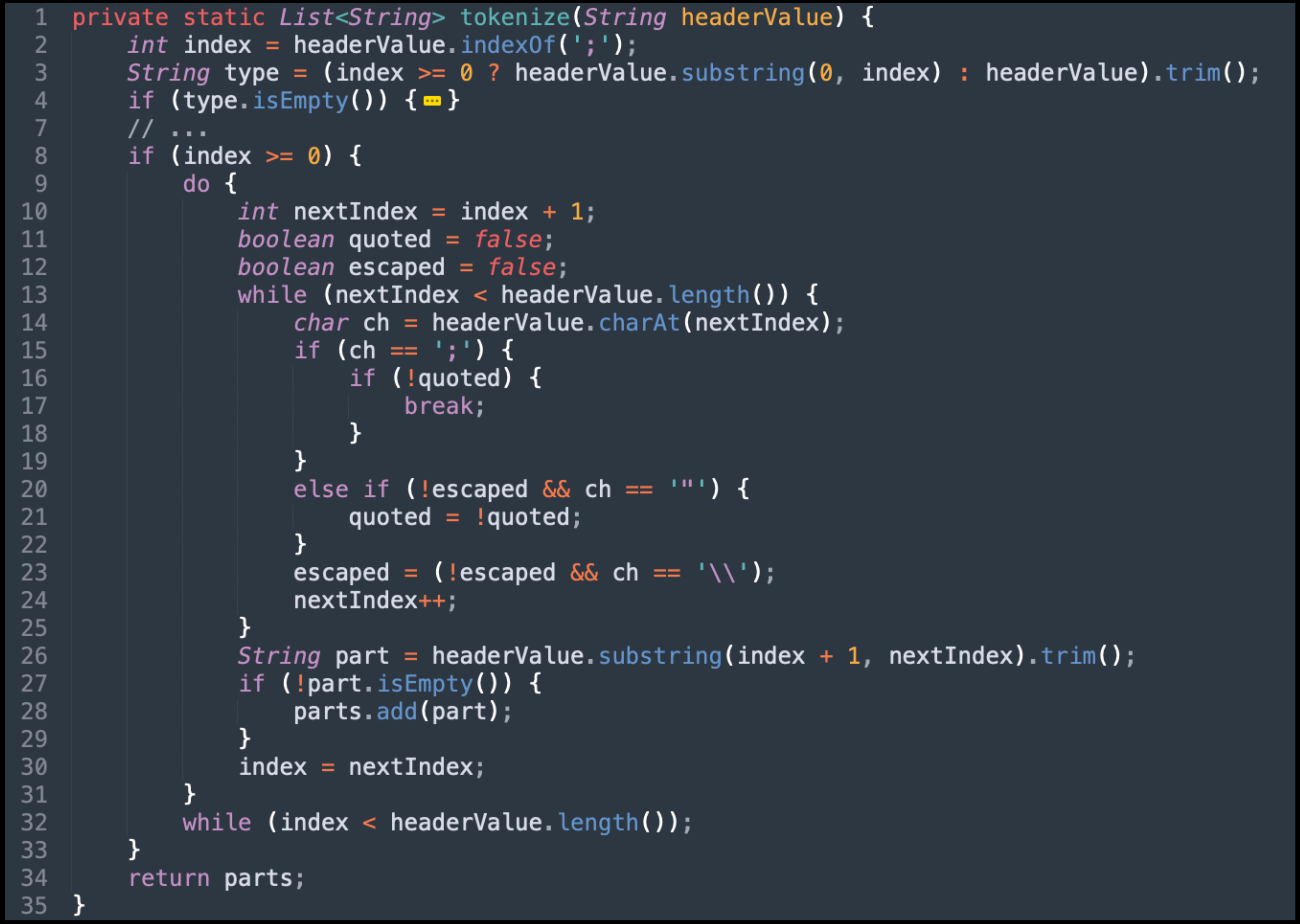
Example Input:

Content-Disposition: form-data; ↩
name=“abc”; ↩
filename=“a.txt”;

Read Server Error - part 2

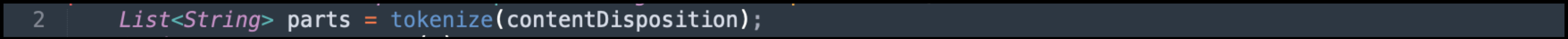


```java
    List<String> parts = tokenize(contentDisposition);
```

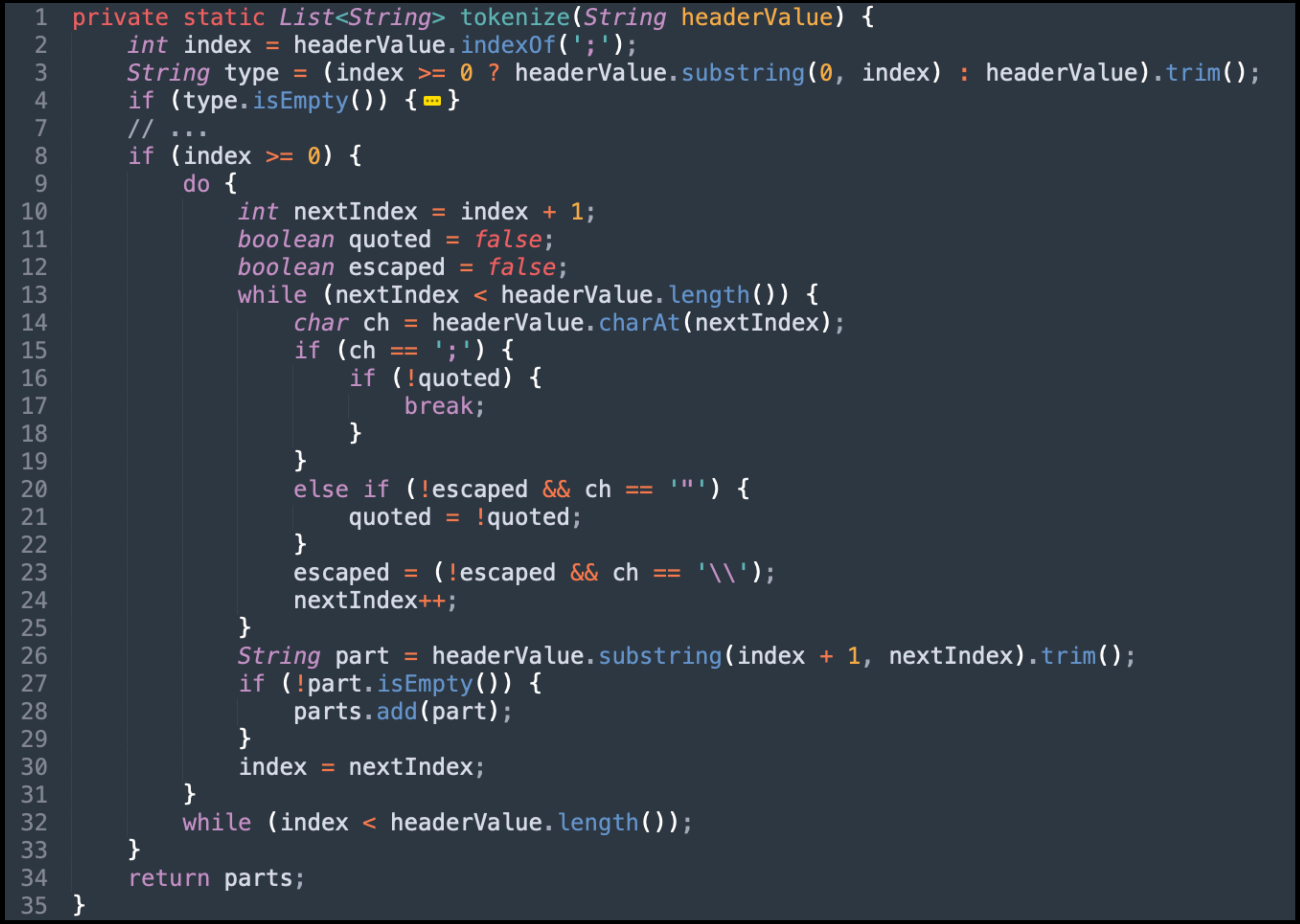


```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {…}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Read Server Error - part 2



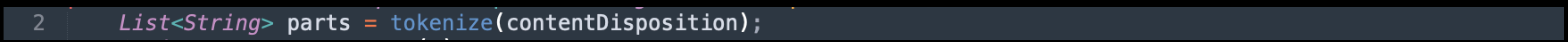
```java
    List<String> parts = tokenize(contentDisposition);
```



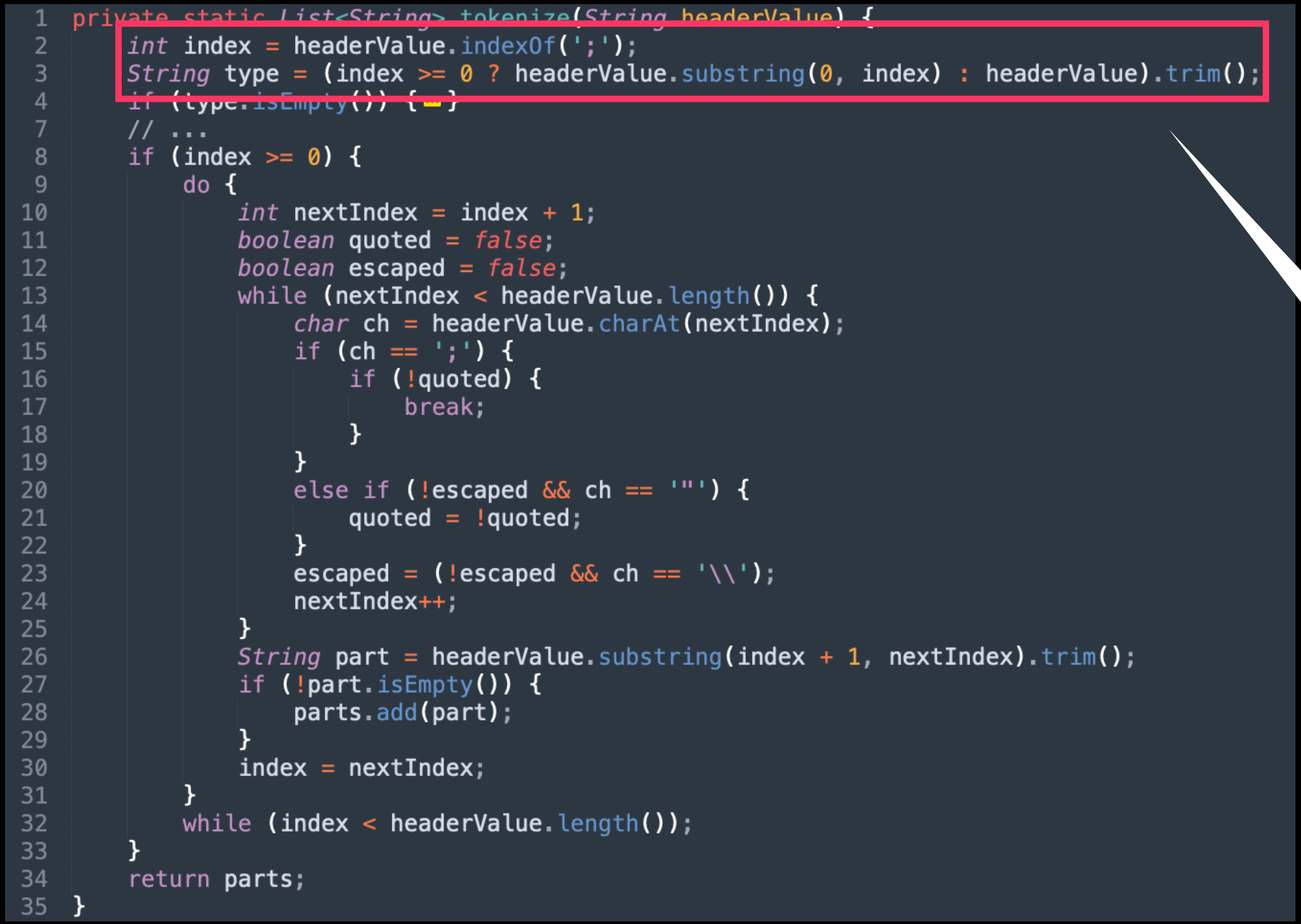
```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {...}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name="abc"; ↩
filename="a.txt";

Read Server Error - part 2



```java
    List<String> parts = tokenize(contentDisposition);
```

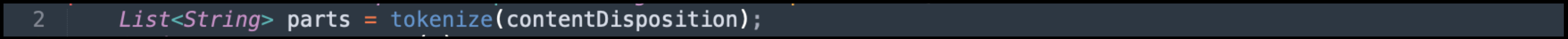


```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {...}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

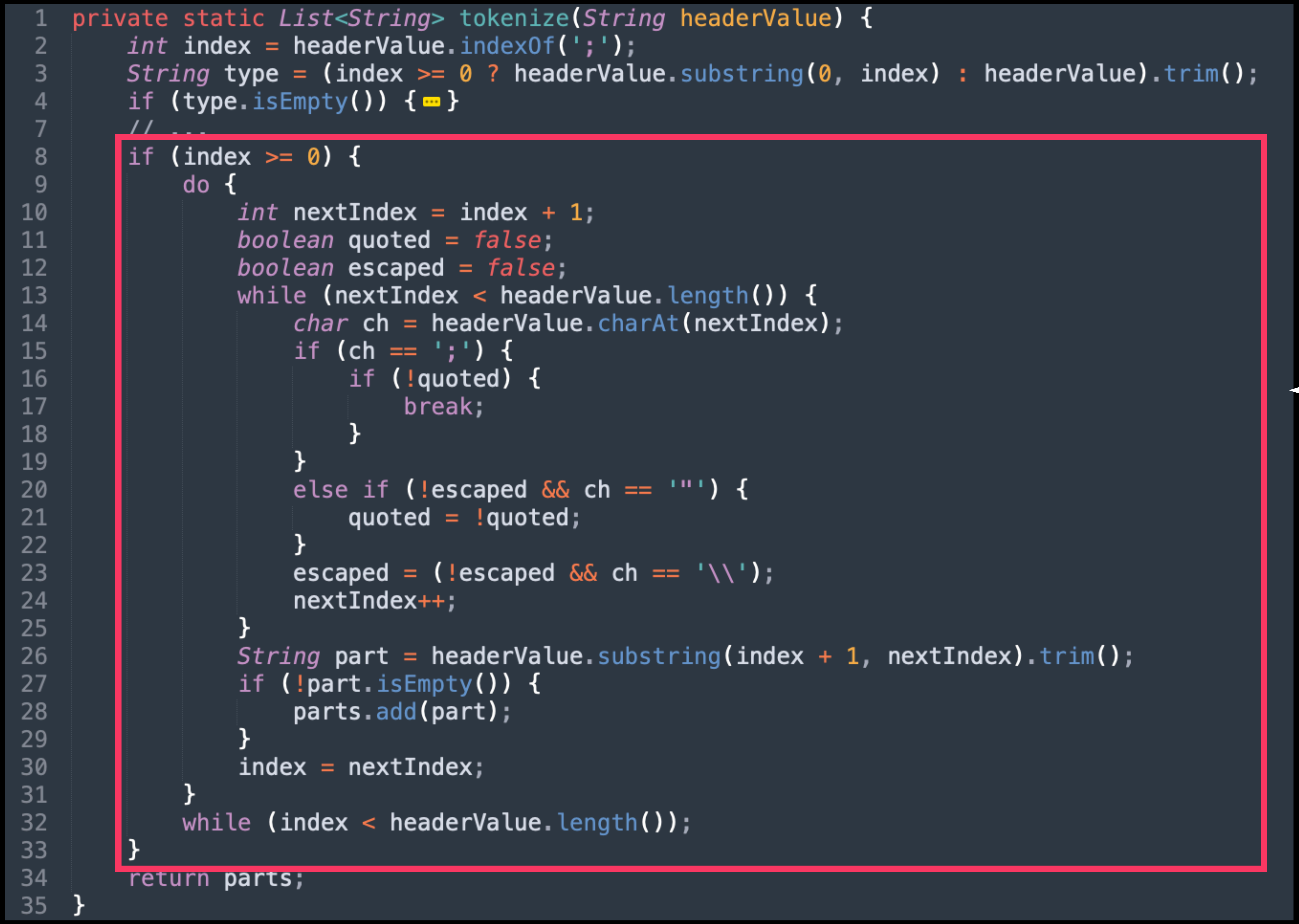
Content-Disposition: form-data; ↩
name="abc"; ↩
filename="a.txt";

Extract Content-Disposition type
(There are form-data, attachment, etc.)

Read Server Error - part 2



```
    List<String> parts = tokenize(contentDisposition);
```

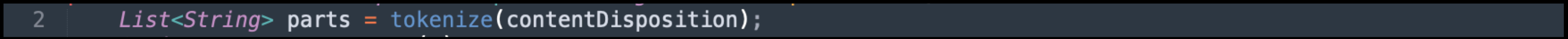


```
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {…}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

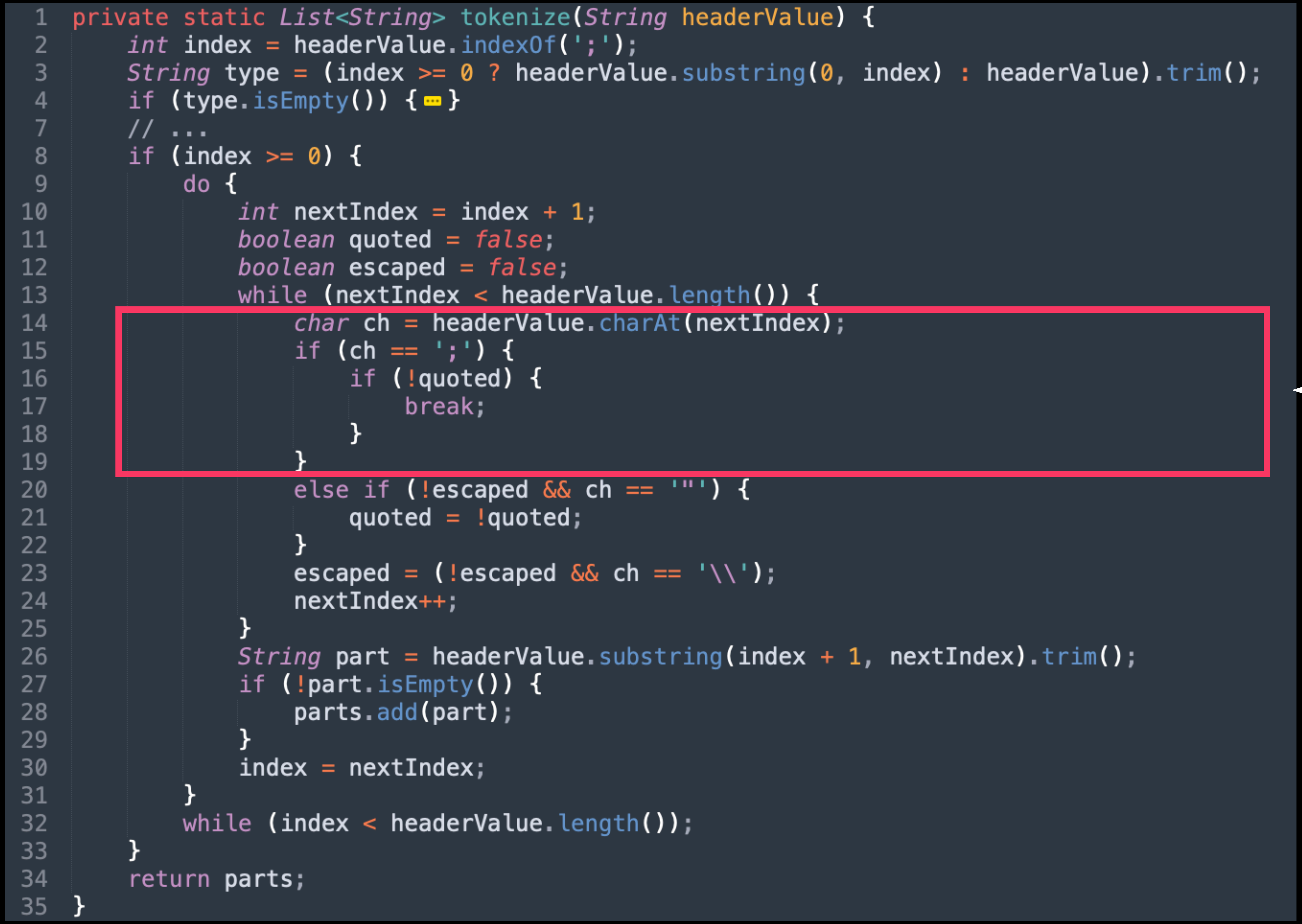
Content-Disposition: form-data; ↩
name=“abc”; ↩
filename=“a.txt”;

Split data into Key-Value groups.

Read Server Error - part 2



```
    List<String> parts = tokenize(contentDisposition);
```

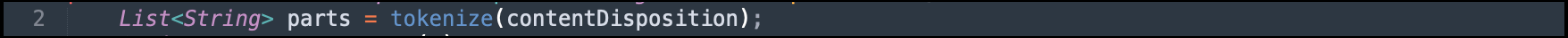


```
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {...}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

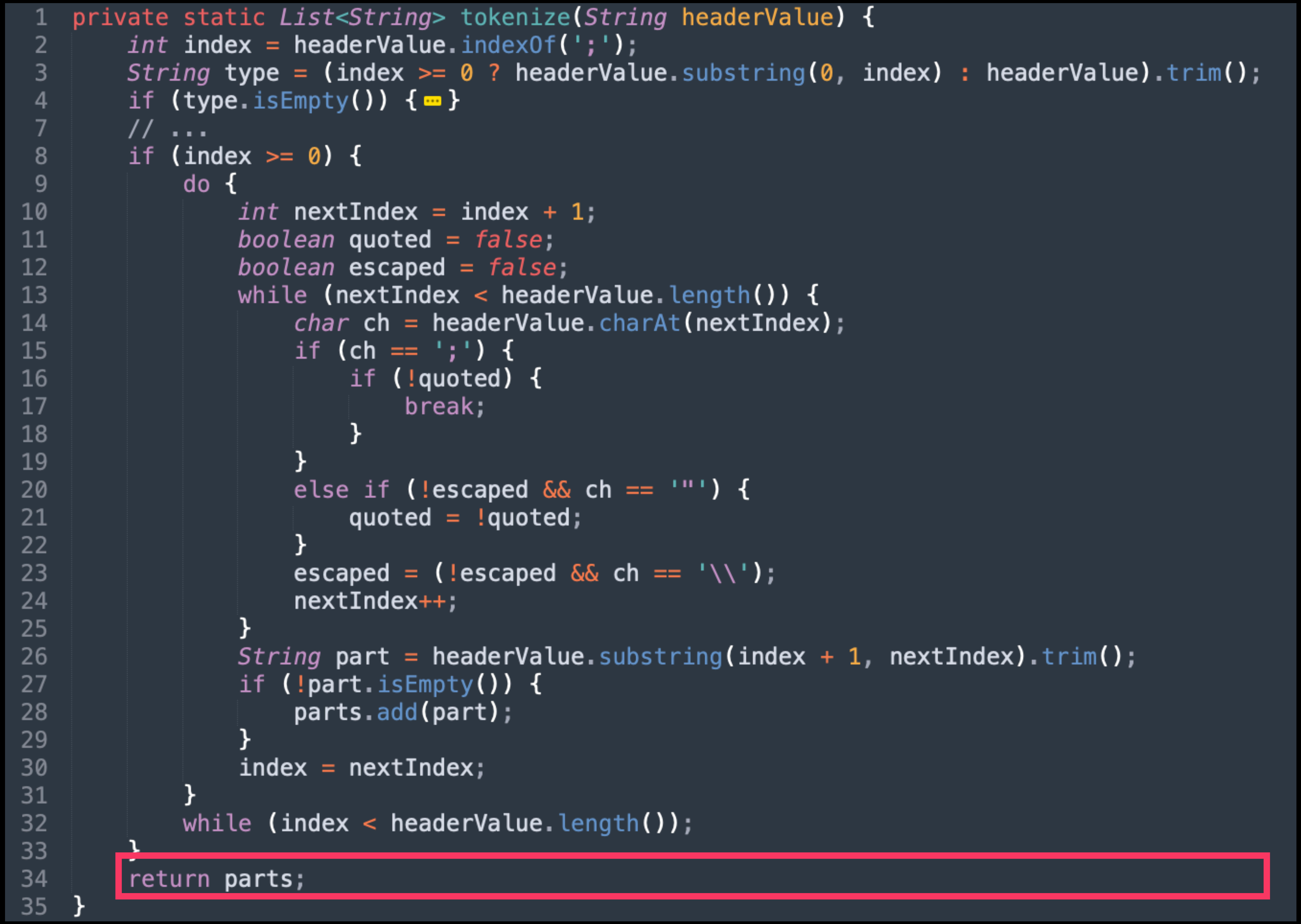
Content-Disposition: form-data; ↩
name="abc"; ↩
filename="a.txt";

If a letter ; is found, the letters up to that index are considered as one group.

Read Server Error - part 2



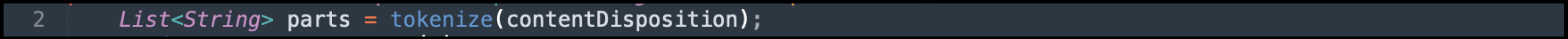
```java
    List<String> parts = tokenize(contentDisposition);
```



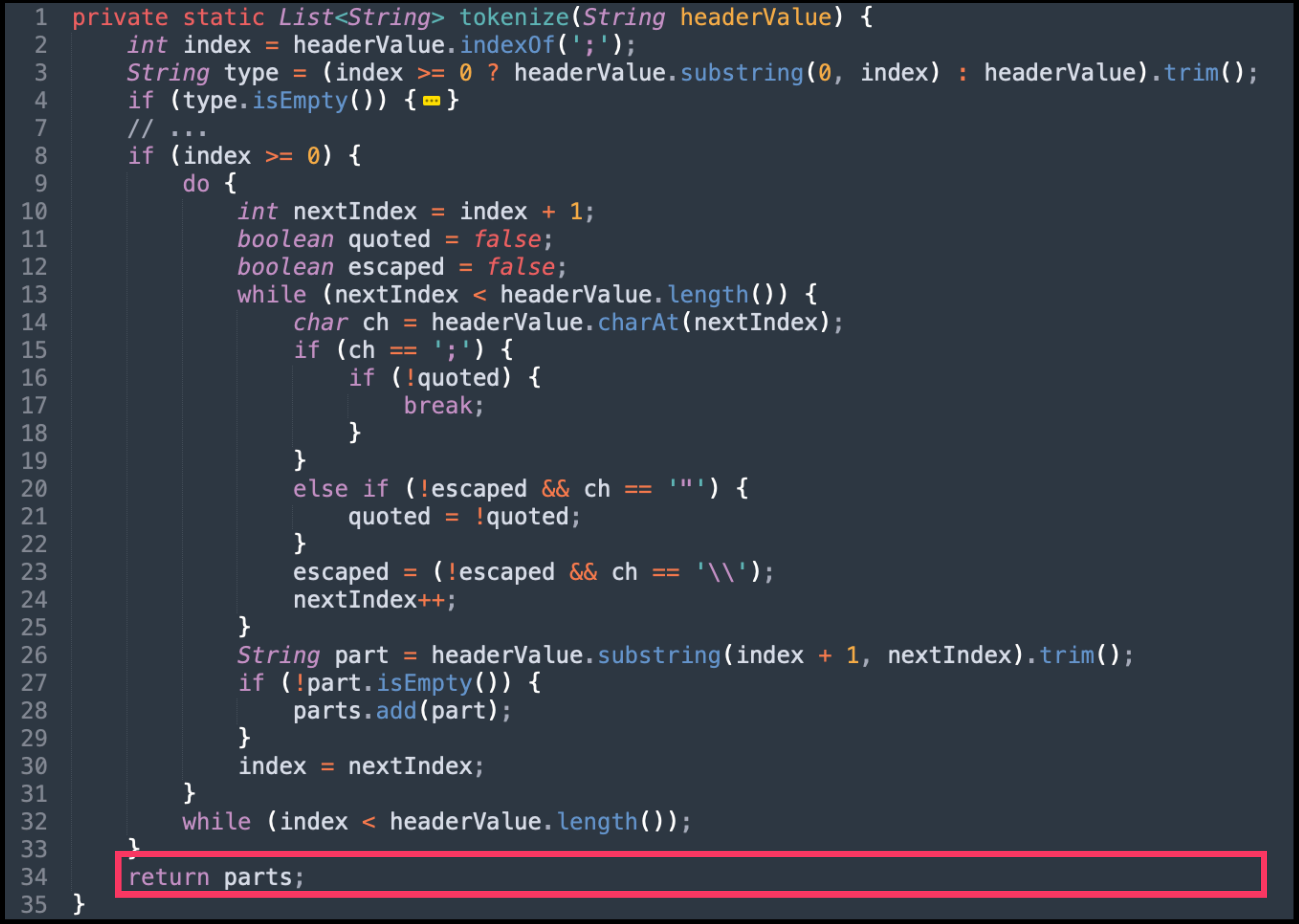
```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {…}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name="abc"; ↩
filename="a.txt";

Read Server Error - part 2



```java
    List<String> parts = tokenize(contentDisposition);
```



```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {…}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name="abc"; ↩
filename="a.txt";

↓

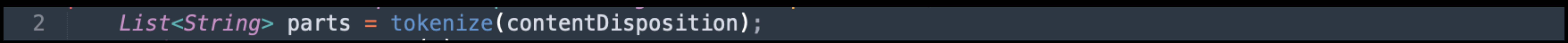
Return:

[0]: **Content-Disposition: form-data**

[1]: **name="abc"**

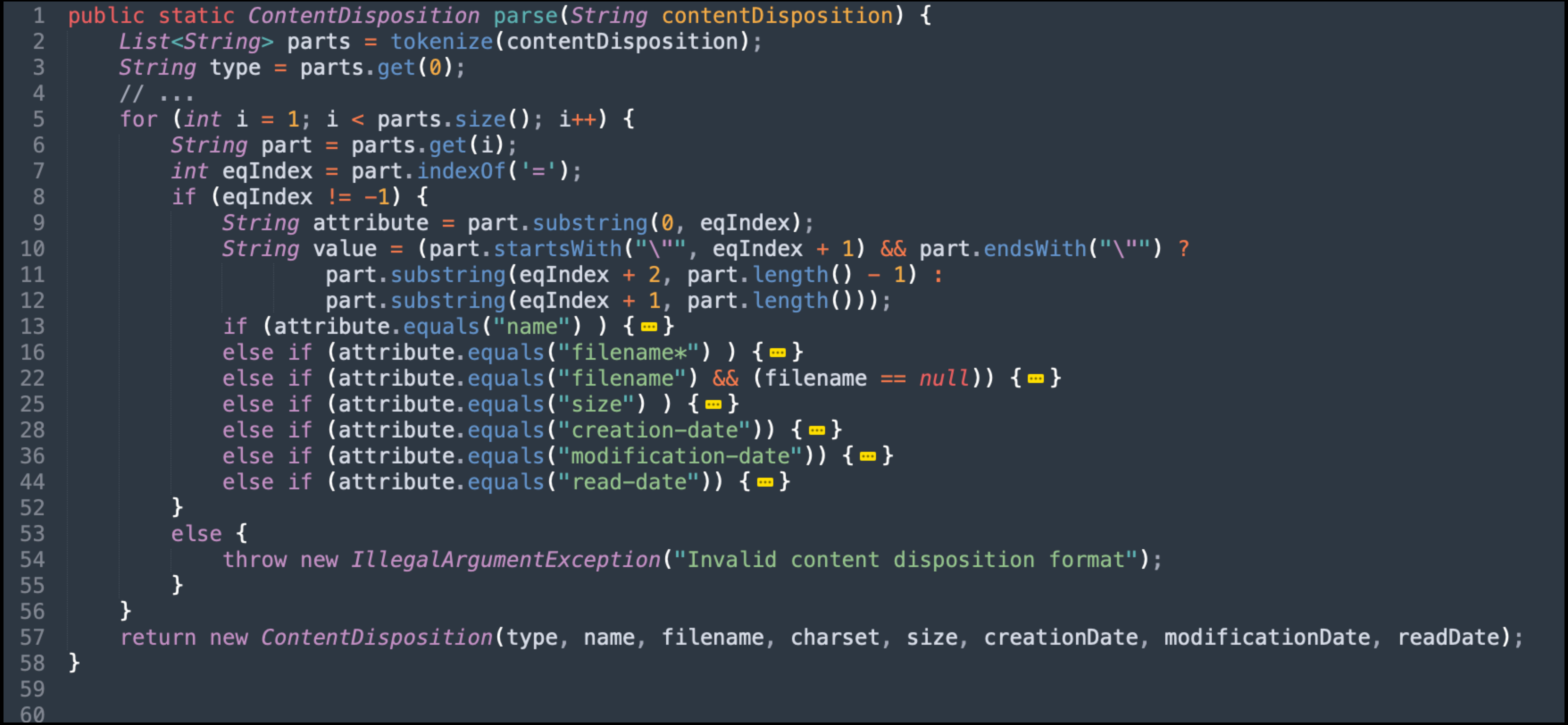
[2]: **filename="a.txt"**

Read Server Error - part 2



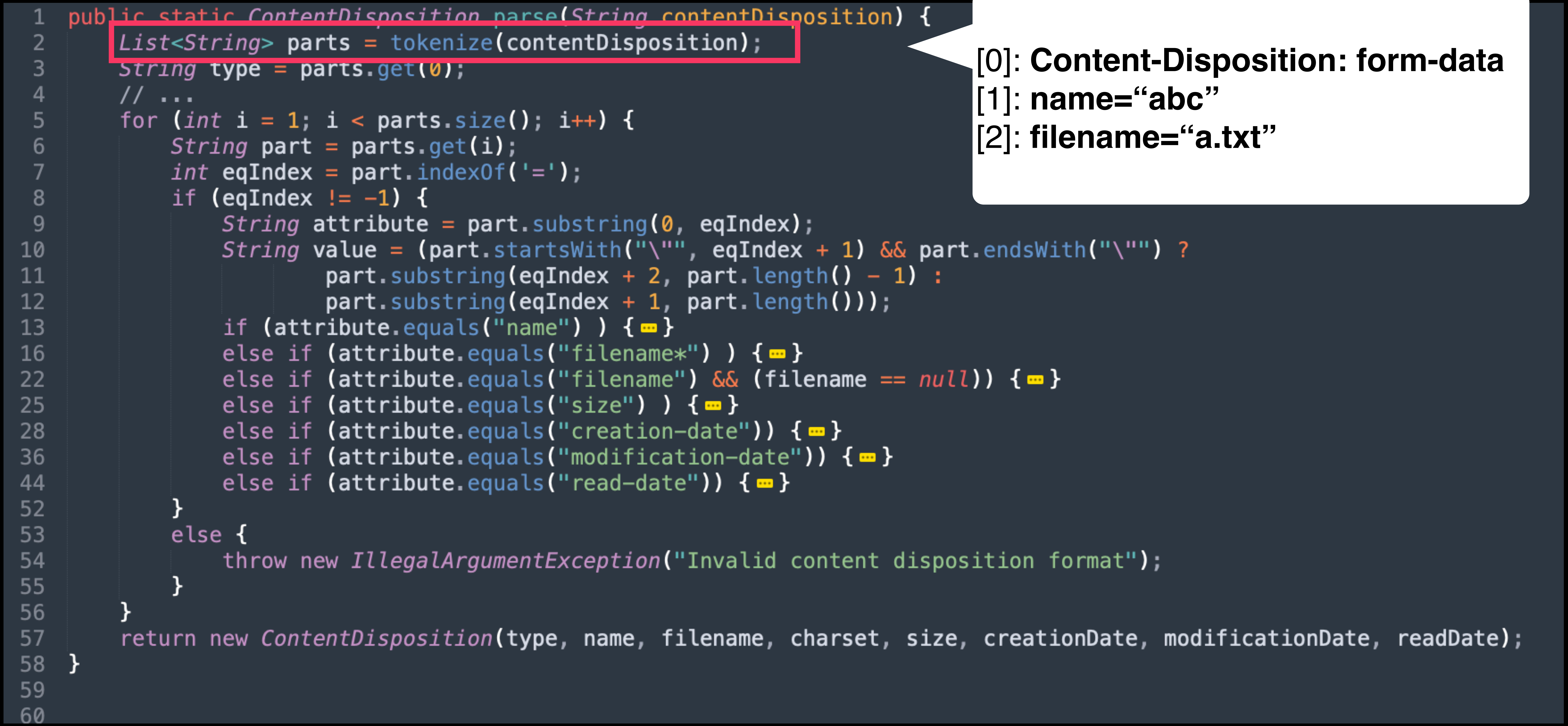
```
    List<String> parts = tokenize(contentDisposition);
```

Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

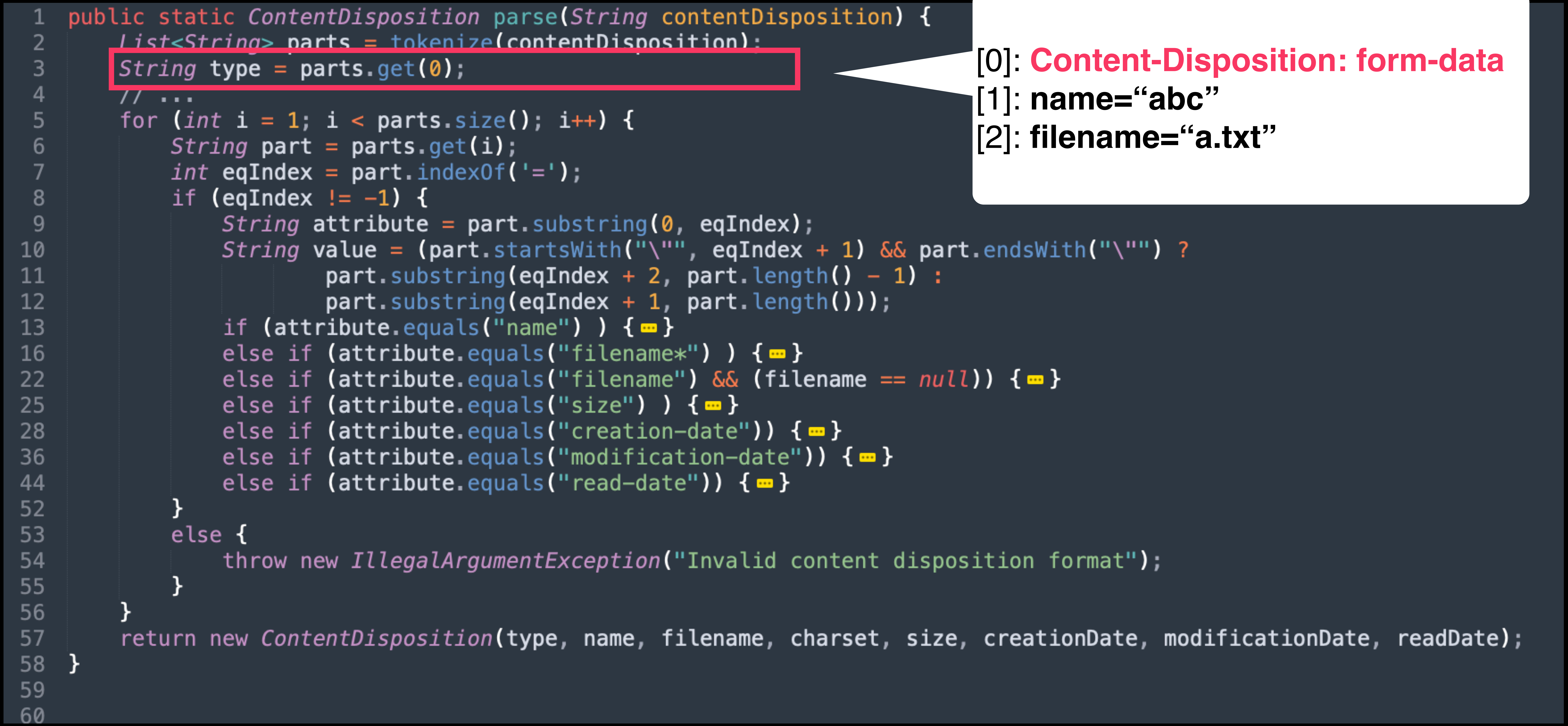
Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
  List<String> parts = tokenize(contentDisposition);
  String type = parts.get(0);
  // ...
  for (int i = 1; i < parts.size(); i++) {
    String part = parts.get(i);
    int eqIndex = part.indexOf('=');
    if (eqIndex != -1) {
      String attribute = part.substring(0, eqIndex);
      String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
          part.substring(eqIndex + 2, part.length() - 1) :
          part.substring(eqIndex + 1, part.length()));
      if (attribute.equals("name") ) {⋯}
      else if (attribute.equals("filename*") ) {⋯}
      else if (attribute.equals("filename") && (filename == null)) {⋯}
      else if (attribute.equals("size") ) {⋯}
      else if (attribute.equals("creation-date")) {⋯}
      else if (attribute.equals("modification-date")) {⋯}
      else if (attribute.equals("read-date")) {⋯}
    }
    else {
      throw new IllegalArgumentException("Invalid content disposition format");
    }
  }
  return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="abc"**
[2]: **filename="a.txt"**

Read Server Error - part 2



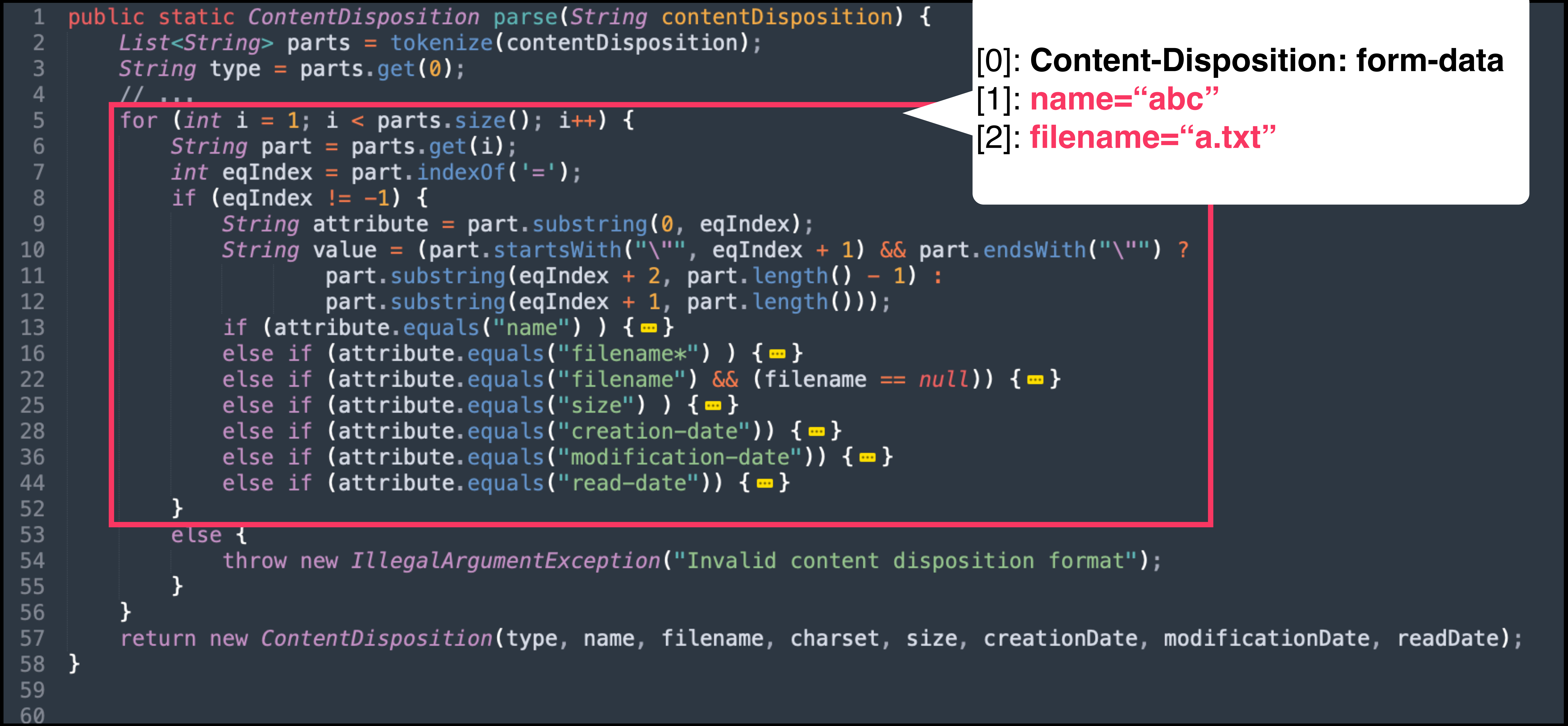
```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**

[1]: **name="abc"**

[2]: **filename="a.txt"**

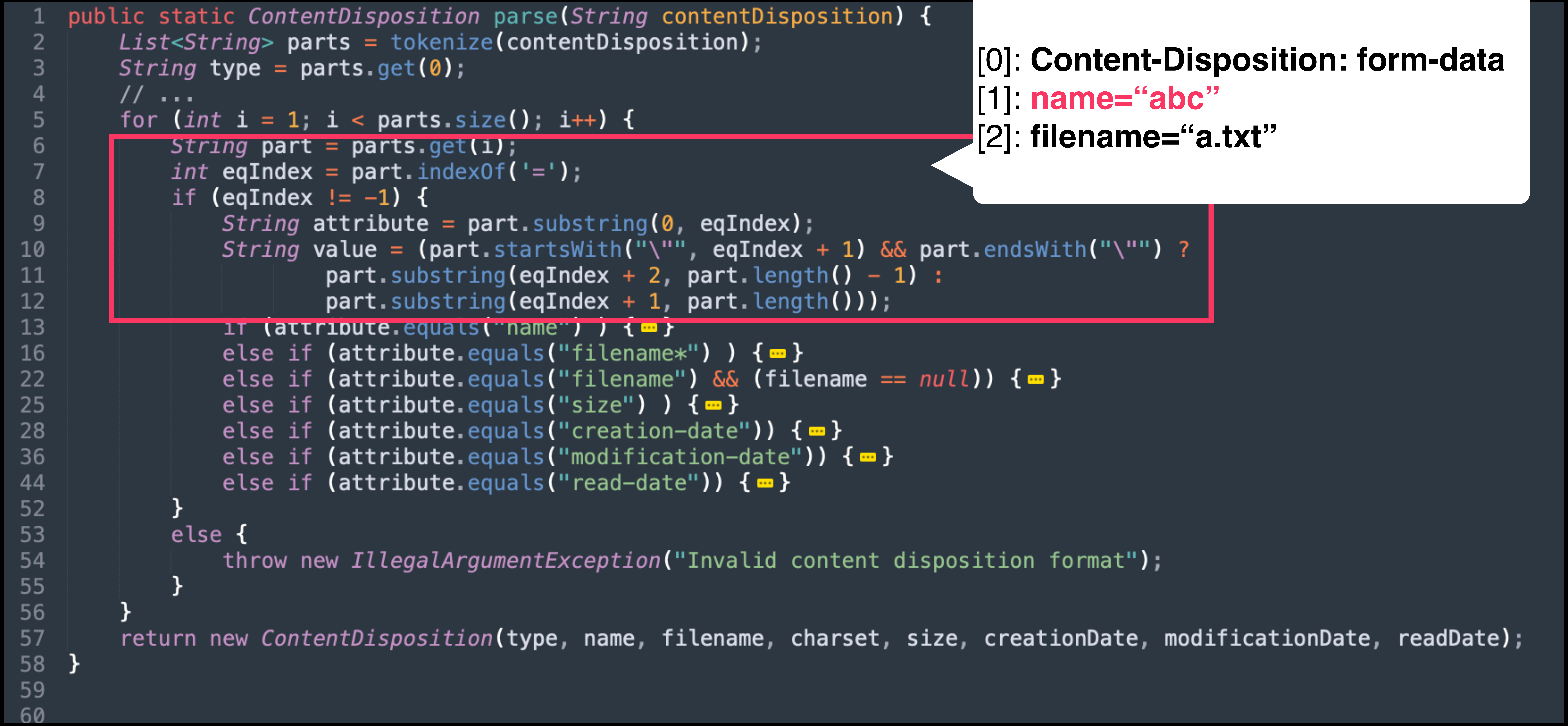
Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {…}
            else if (attribute.equals("filename*") ) {…}
            else if (attribute.equals("filename") && (filename == null)) {…}
            else if (attribute.equals("size") ) {…}
            else if (attribute.equals("creation-date")) {…}
            else if (attribute.equals("modification-date")) {…}
            else if (attribute.equals("read-date")) {…}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="abc"**
[2]: **filename="a.txt"**

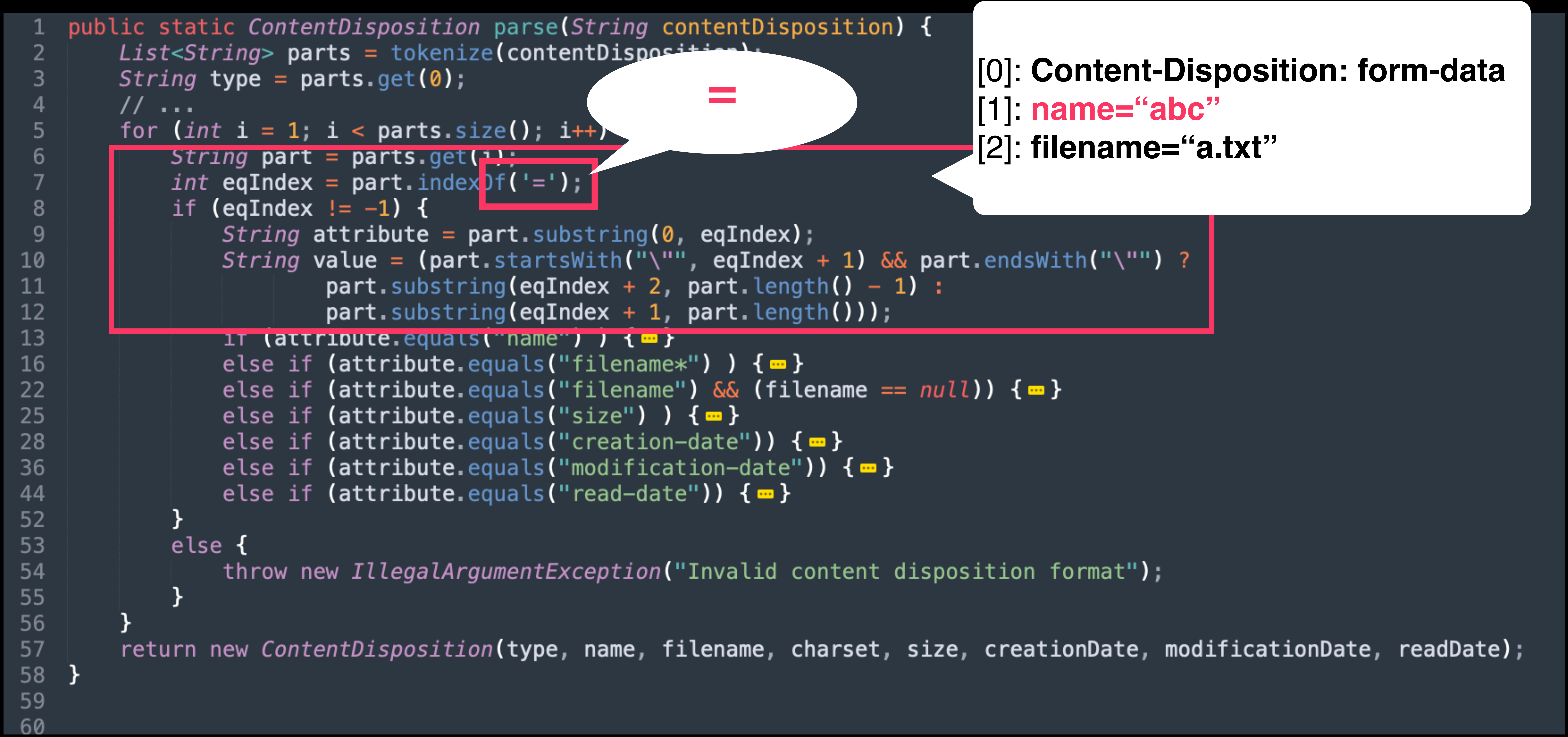
Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="abc"**
[2]: **filename="a.txt"**

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

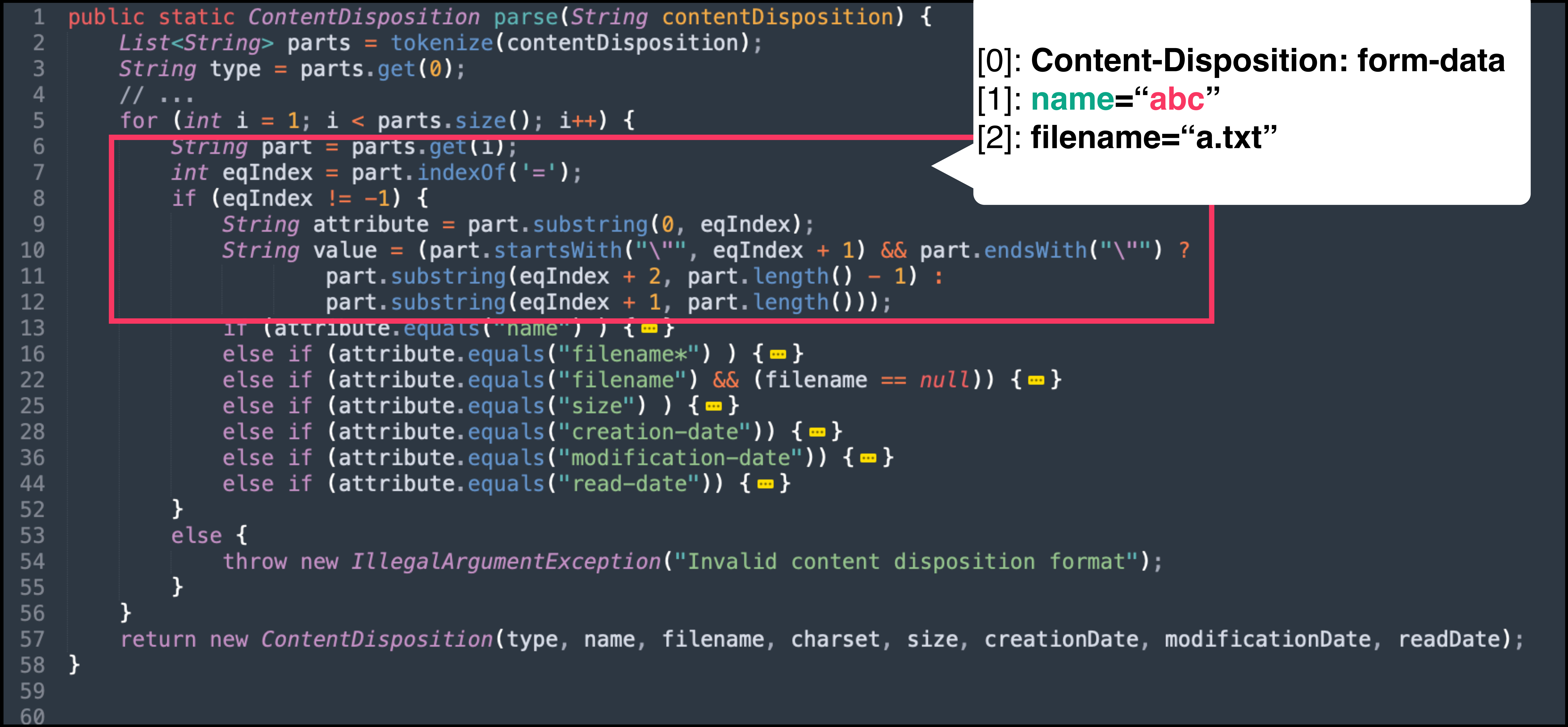
=

[0]: **Content-Disposition: form-data**

[1]: **name="abc"**

[2]: **filename="a.txt"**

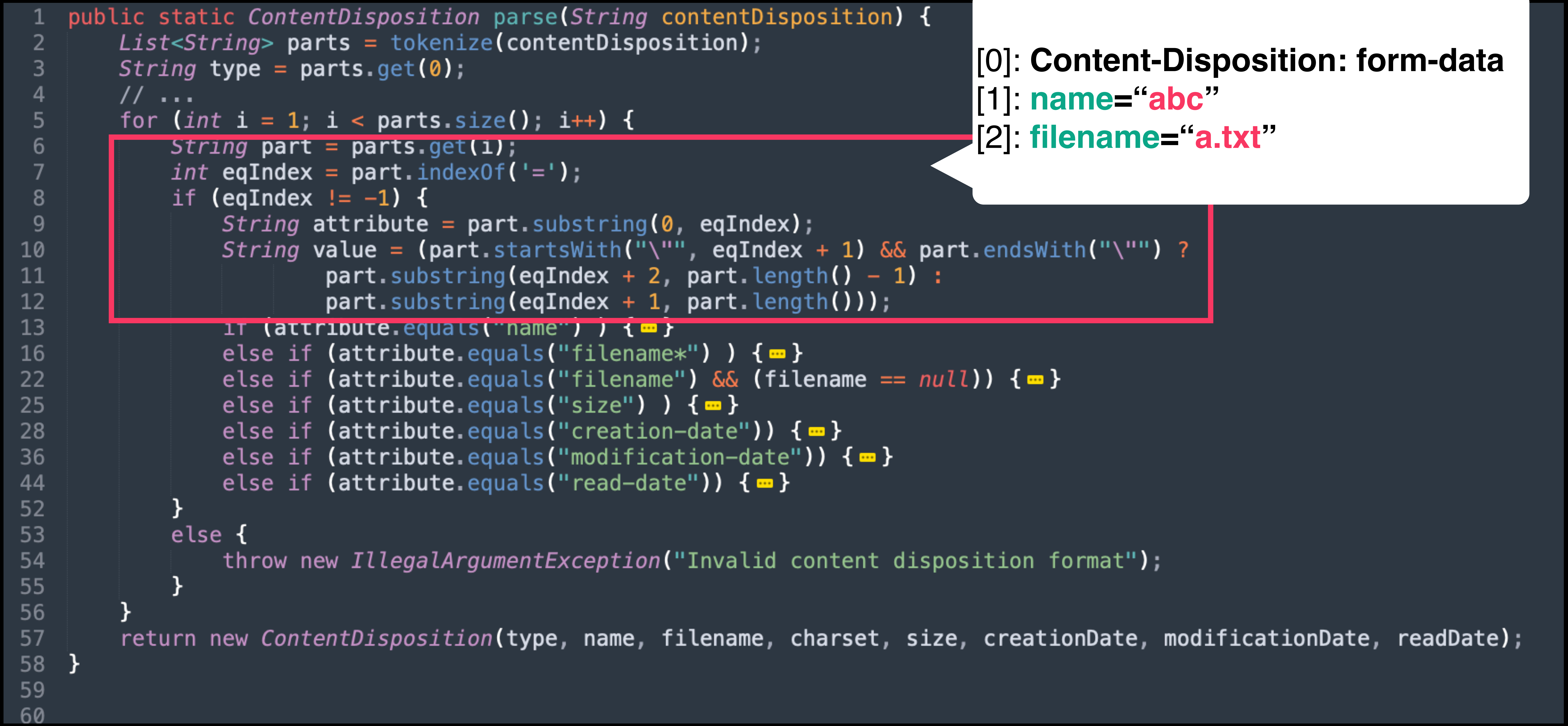
Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name=“abc”**
[2]: **filename=“a.txt”**

Read Server Error - part 2



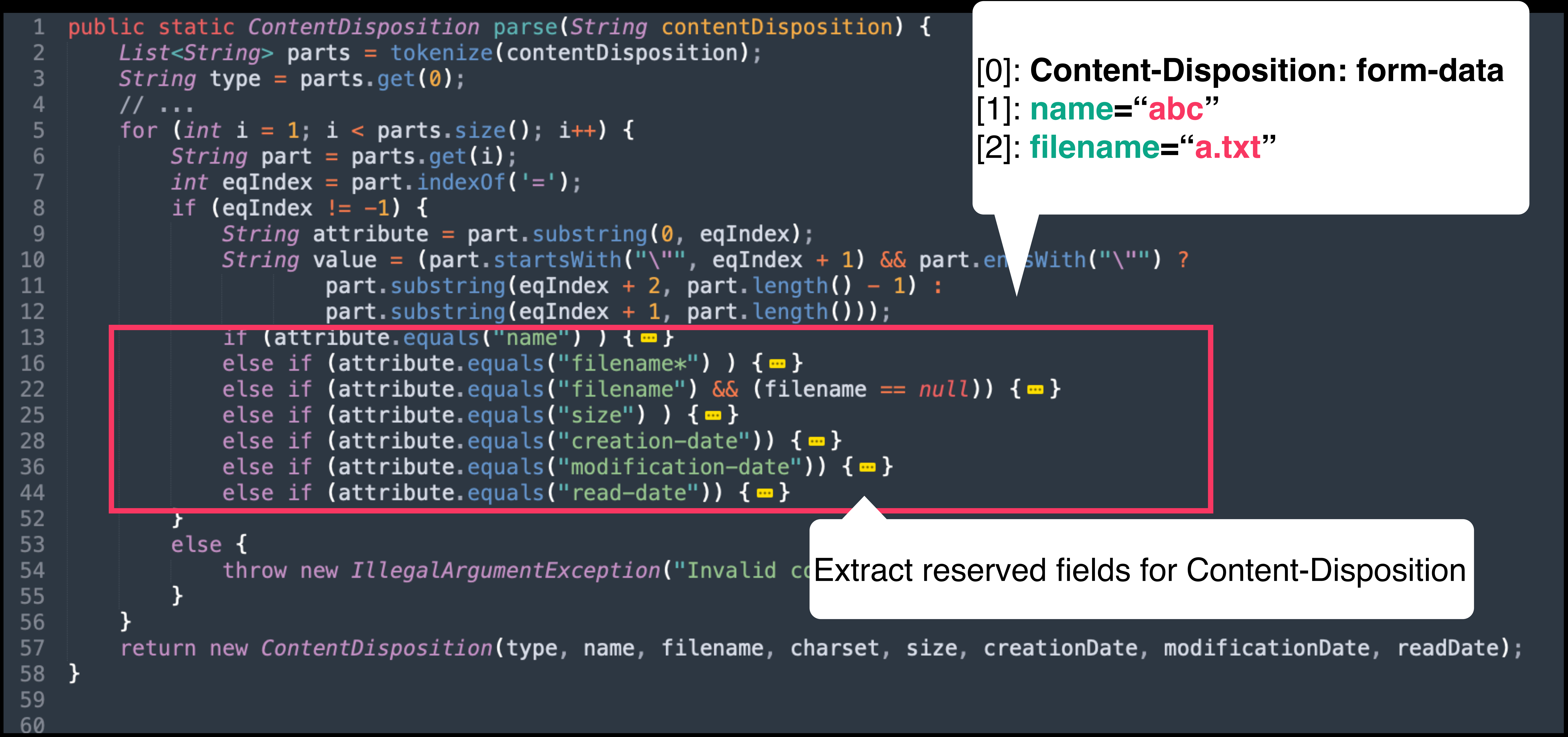
```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**

[1]: **name="abc"**

[2]: **filename="a.txt"**

Read Server Error - part 2

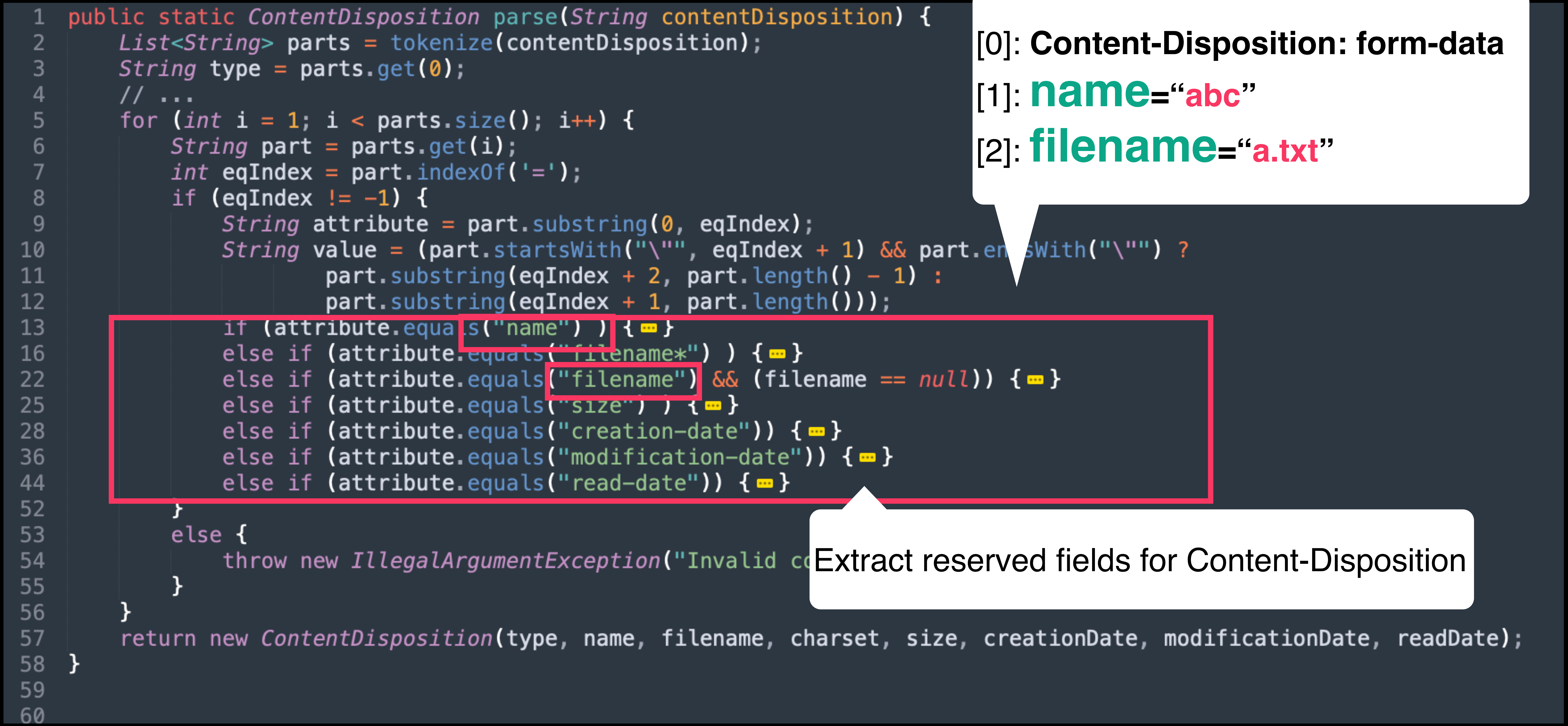


```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.en sWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid co
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="abc"**
[2]: **filename="a.txt"**

Extract reserved fields for Content-Disposition

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid co
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

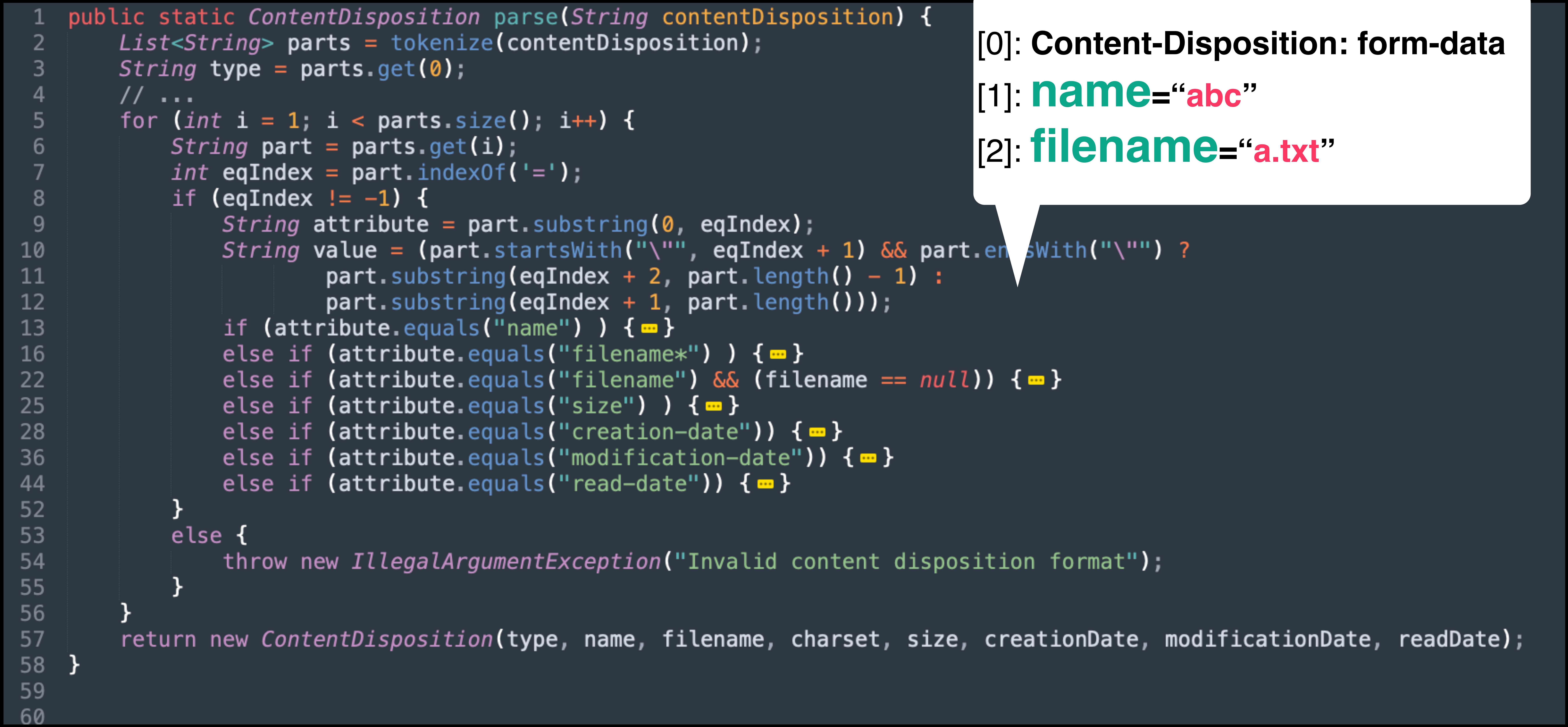
[0]: **Content-Disposition: form-data**

[1]: **name**="**abc**"

[2]: **filename**="**a.txt**"

Extract reserved fields for Content-Disposition

Read Server Error - part 2



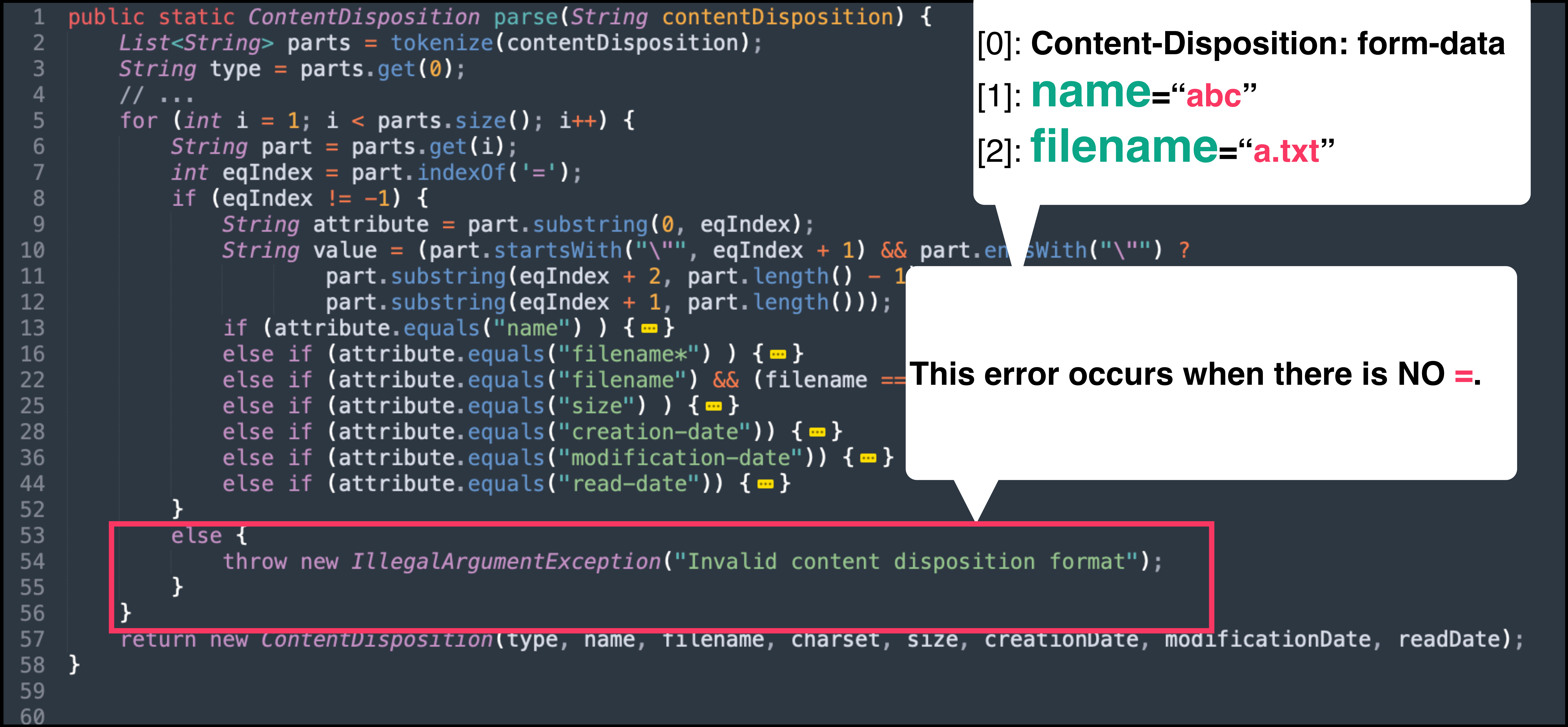
```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.en sWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**

[1]: **name**=“abc”

[2]: **filename**=“a.txt”

Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.en sWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {…}
            else if (attribute.equals("filename*") ) {…}
            else if (attribute.equals("filename") && (filename ==
            else if (attribute.equals("size") ) {…}
            else if (attribute.equals("creation-date")) {…}
            else if (attribute.equals("modification-date")) {…}
            else if (attribute.equals("read-date")) {…}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

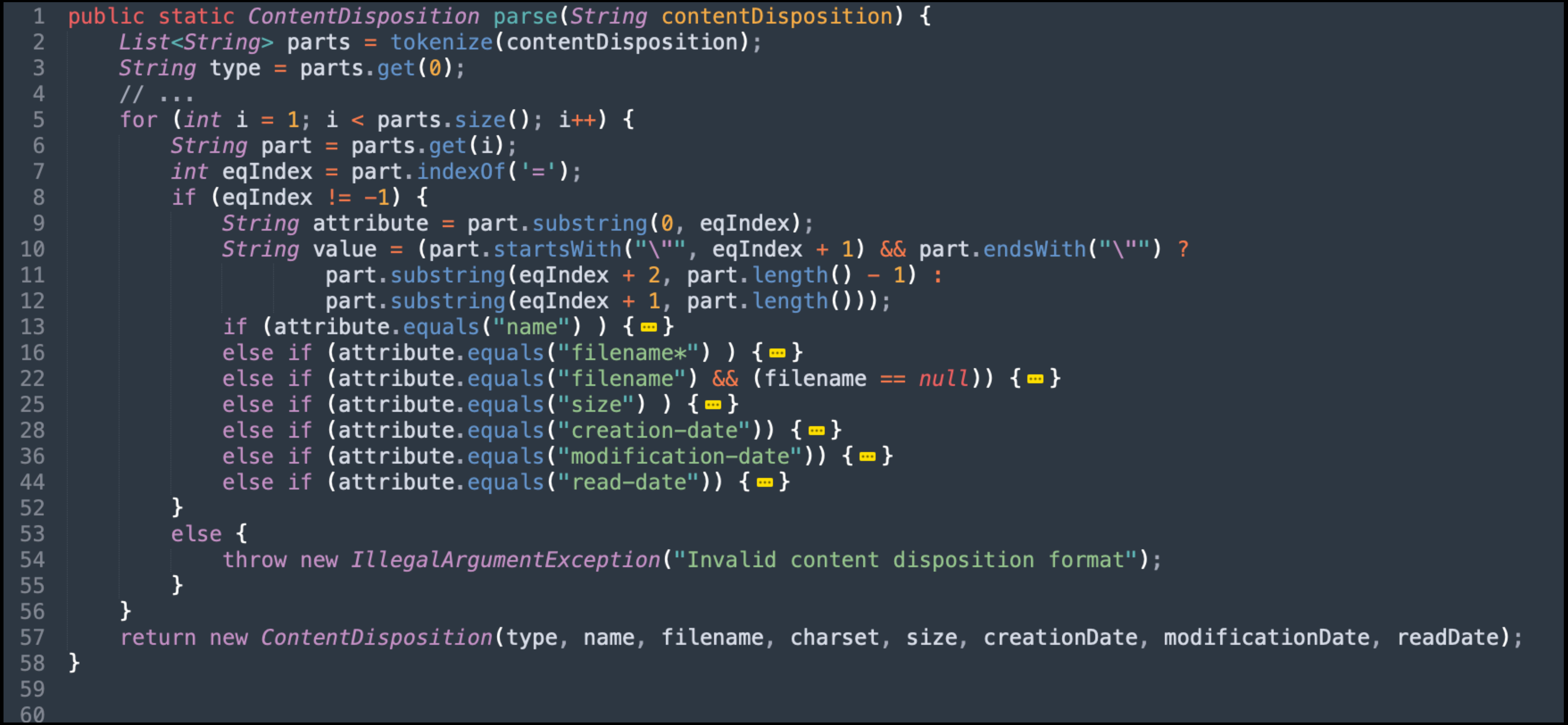
[0]: **Content-Disposition: form-data**

[1]: **name**=“**abc**”

[2]: **filename**=“**a.txt**”

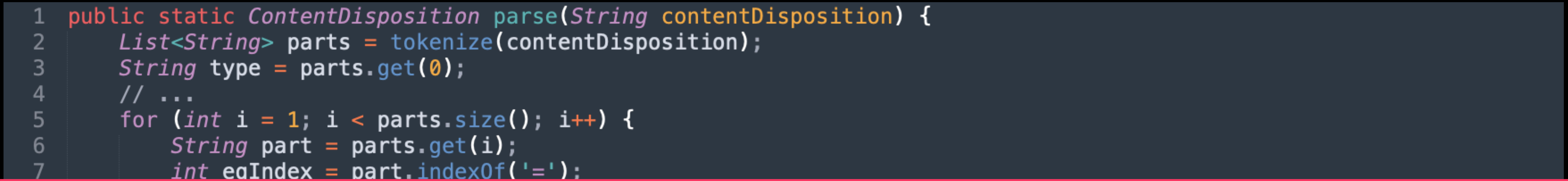
This error occurs when there is NO =.

Read Server Error - part 2

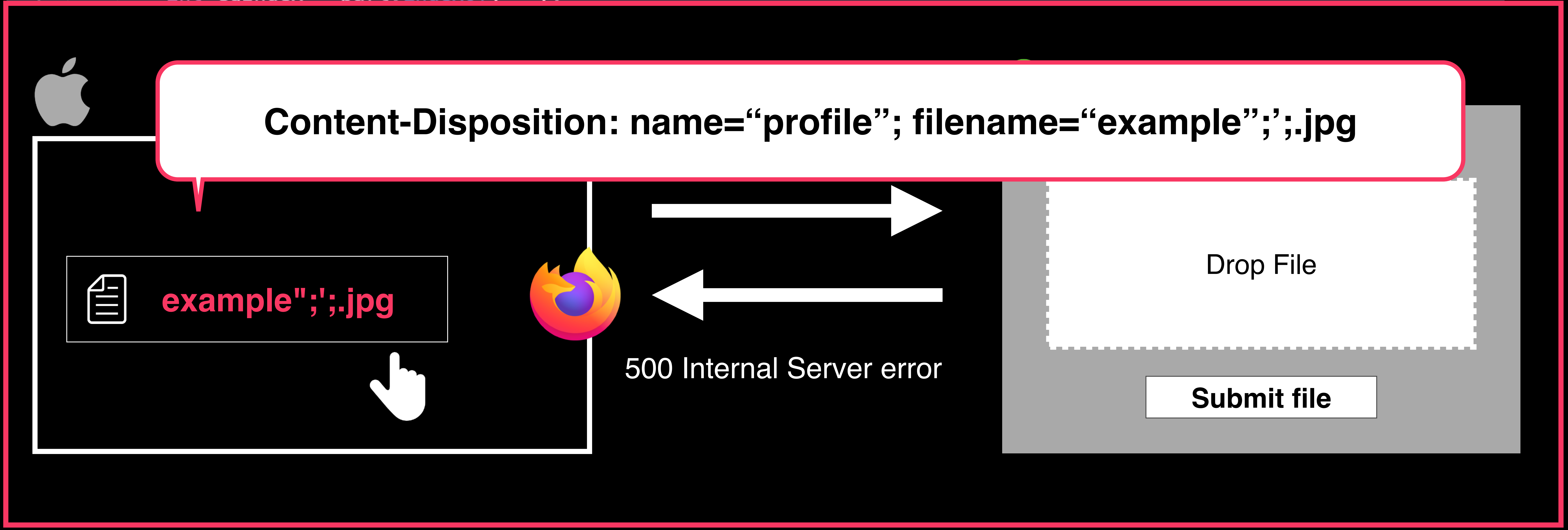


```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {…}
            else if (attribute.equals("filename*") ) {…}
            else if (attribute.equals("filename") && (filename == null)) {…}
            else if (attribute.equals("size") ) {…}
            else if (attribute.equals("creation-date")) {…}
            else if (attribute.equals("modification-date")) {…}
            else if (attribute.equals("read-date")) {…}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
```



Content-Disposition: name=“profile”; filename=“example”;’;.jpg

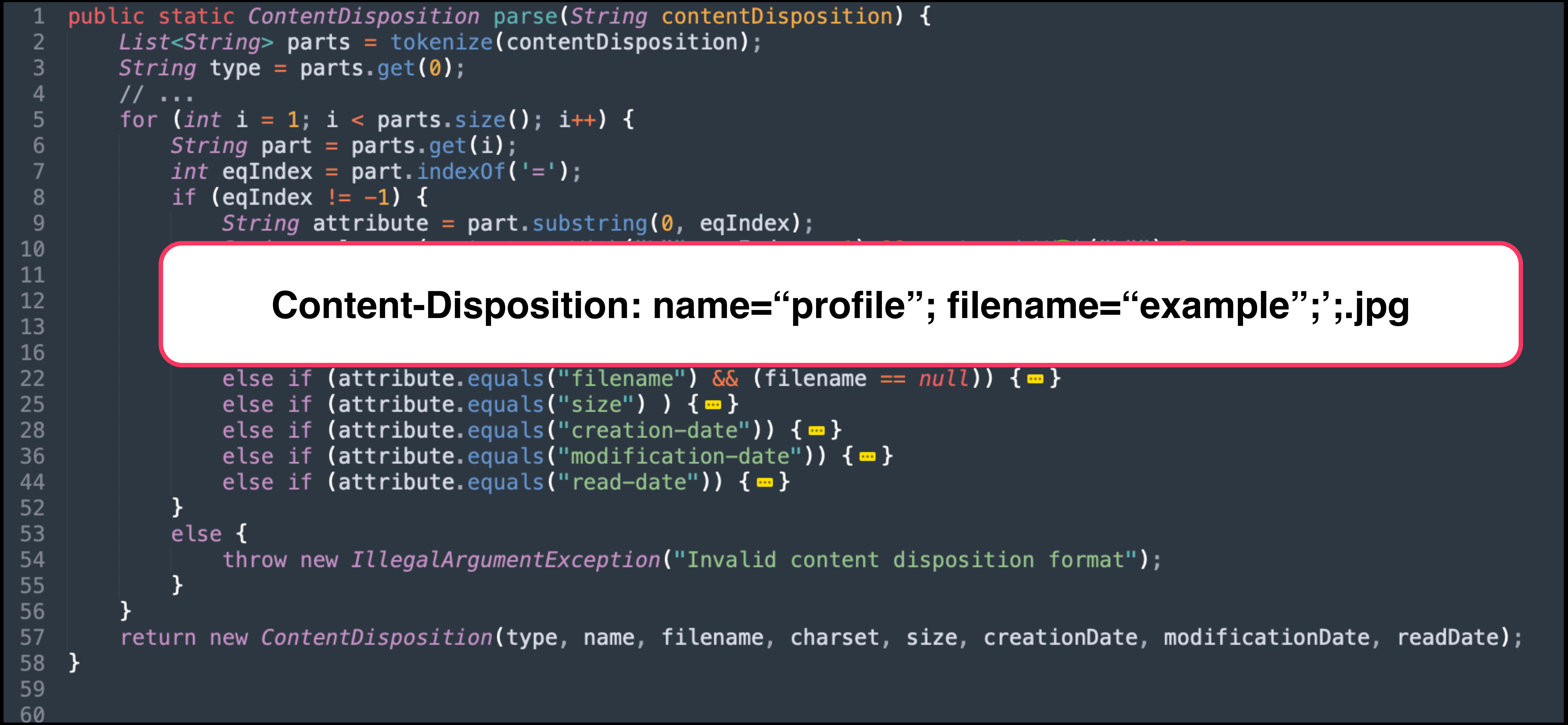
example";';.jpg

500 Internal Server error

Drop File

Submit file

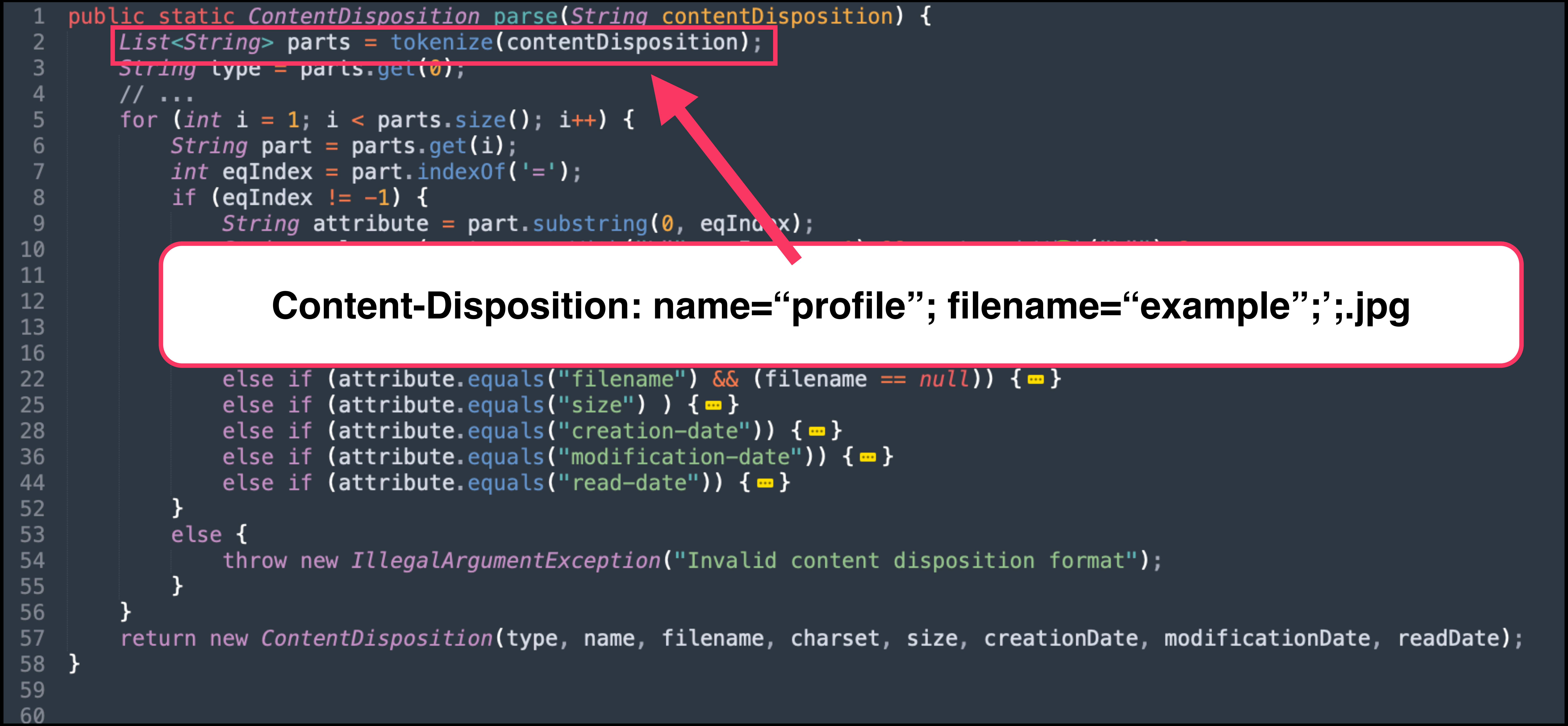
Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);

            else if (attribute.equals("filename") && (filename == null)) {…}
            else if (attribute.equals("size") ) {…}
            else if (attribute.equals("creation-date")) {…}
            else if (attribute.equals("modification-date")) {…}
            else if (attribute.equals("read-date")) {…}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Content-Disposition: name=“profile”; filename=“example”;’;.jpg

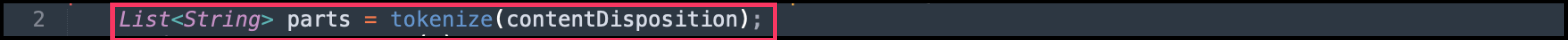
Read Server Error - part 2



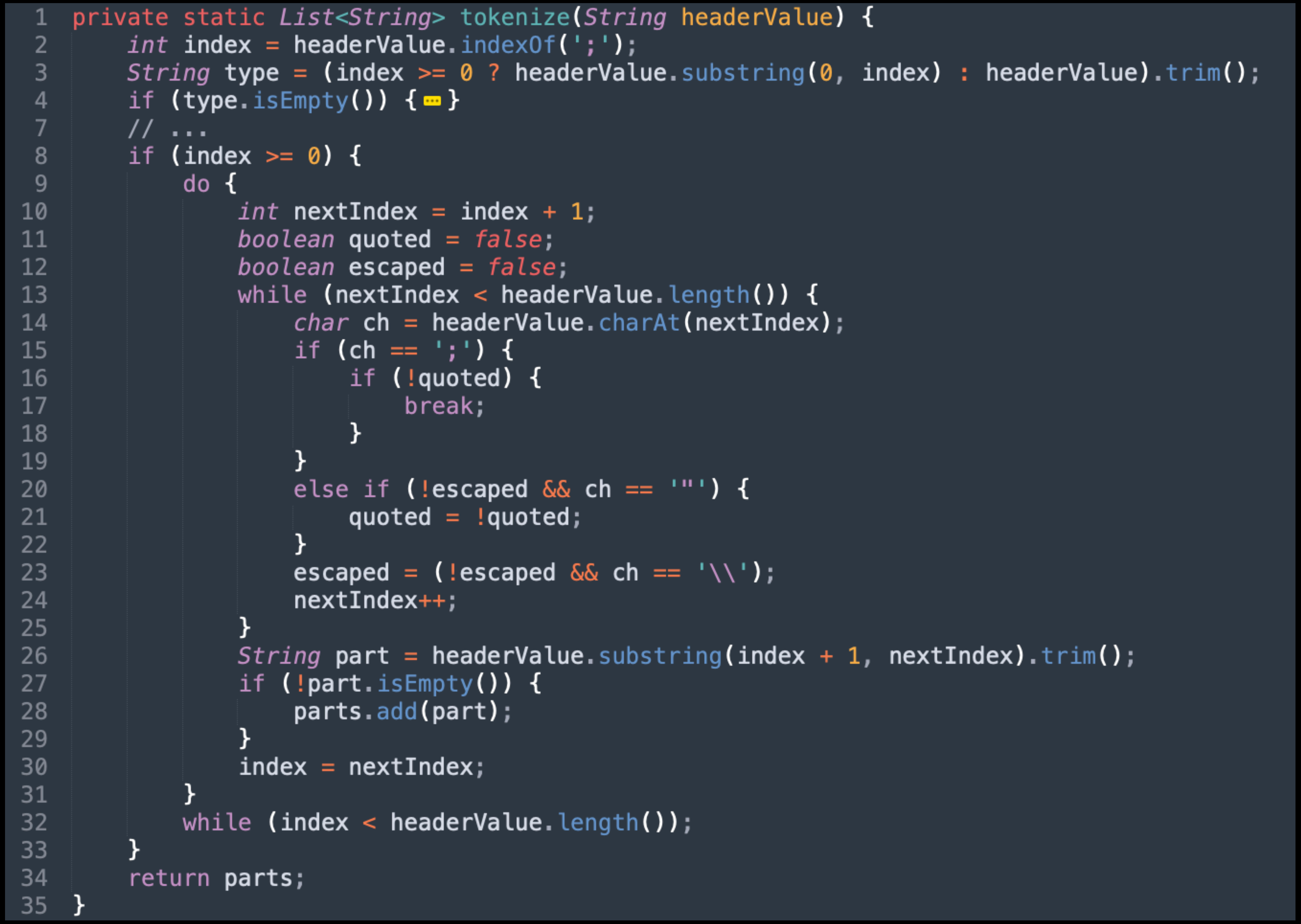
```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            else if (attribute.equals("filename") && (filename == null)) {…}
            else if (attribute.equals("size") ) {…}
            else if (attribute.equals("creation-date")) {…}
            else if (attribute.equals("modification-date")) {…}
            else if (attribute.equals("read-date")) {…}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

Content-Disposition: name=“profile”; filename=“example”;’;.jpg

Read Server Error - part 2

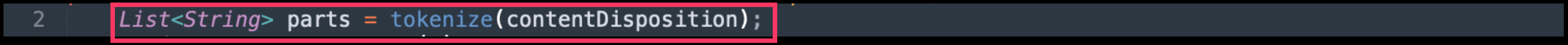


```java
List<String> parts = tokenize(contentDisposition);
```

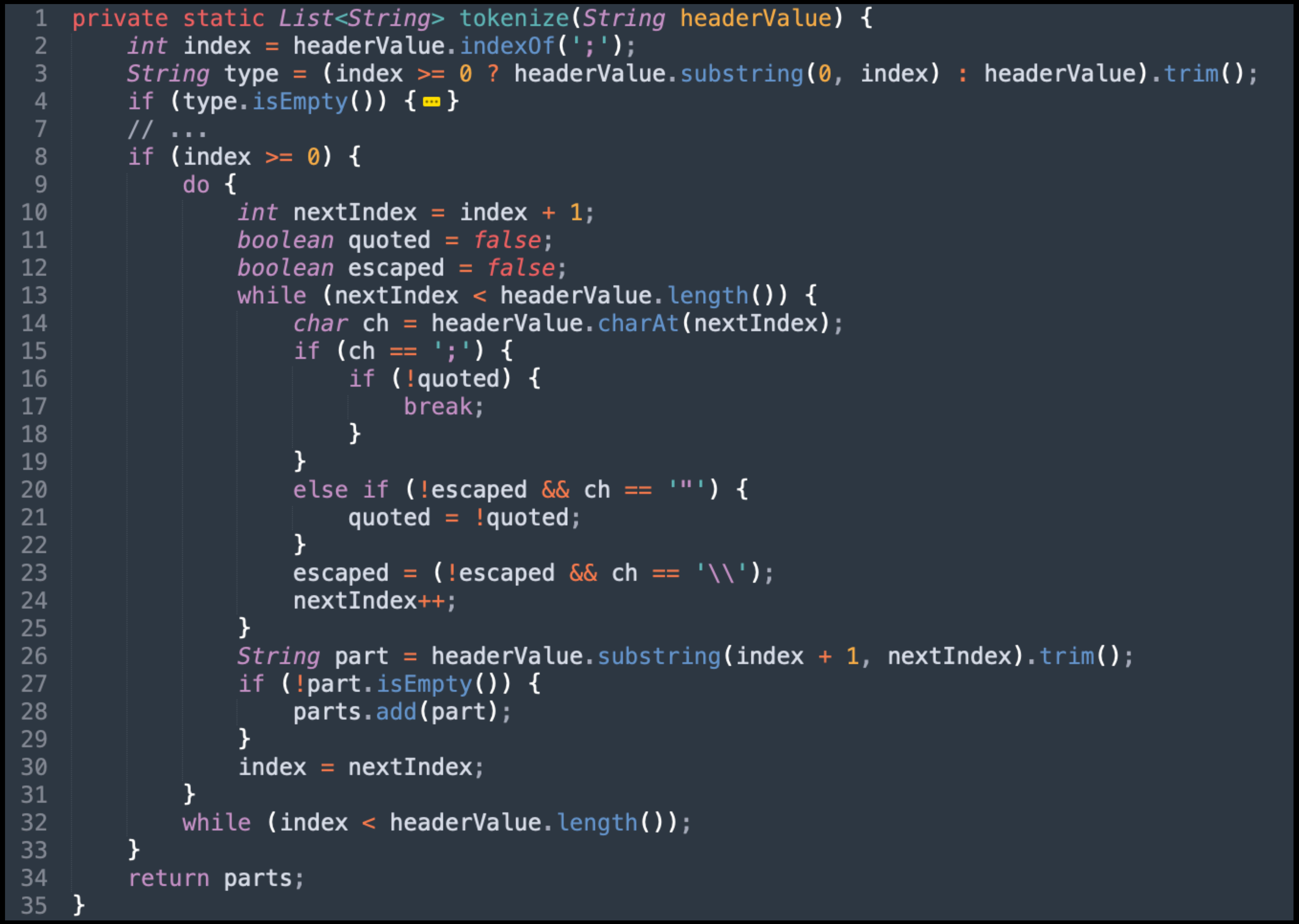


```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {⋯}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Read Server Error - part 2



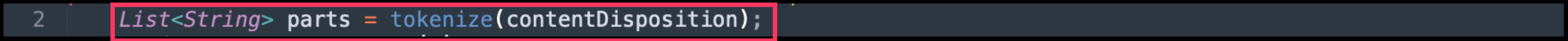
```java
    List<String> parts = tokenize(contentDisposition);
```



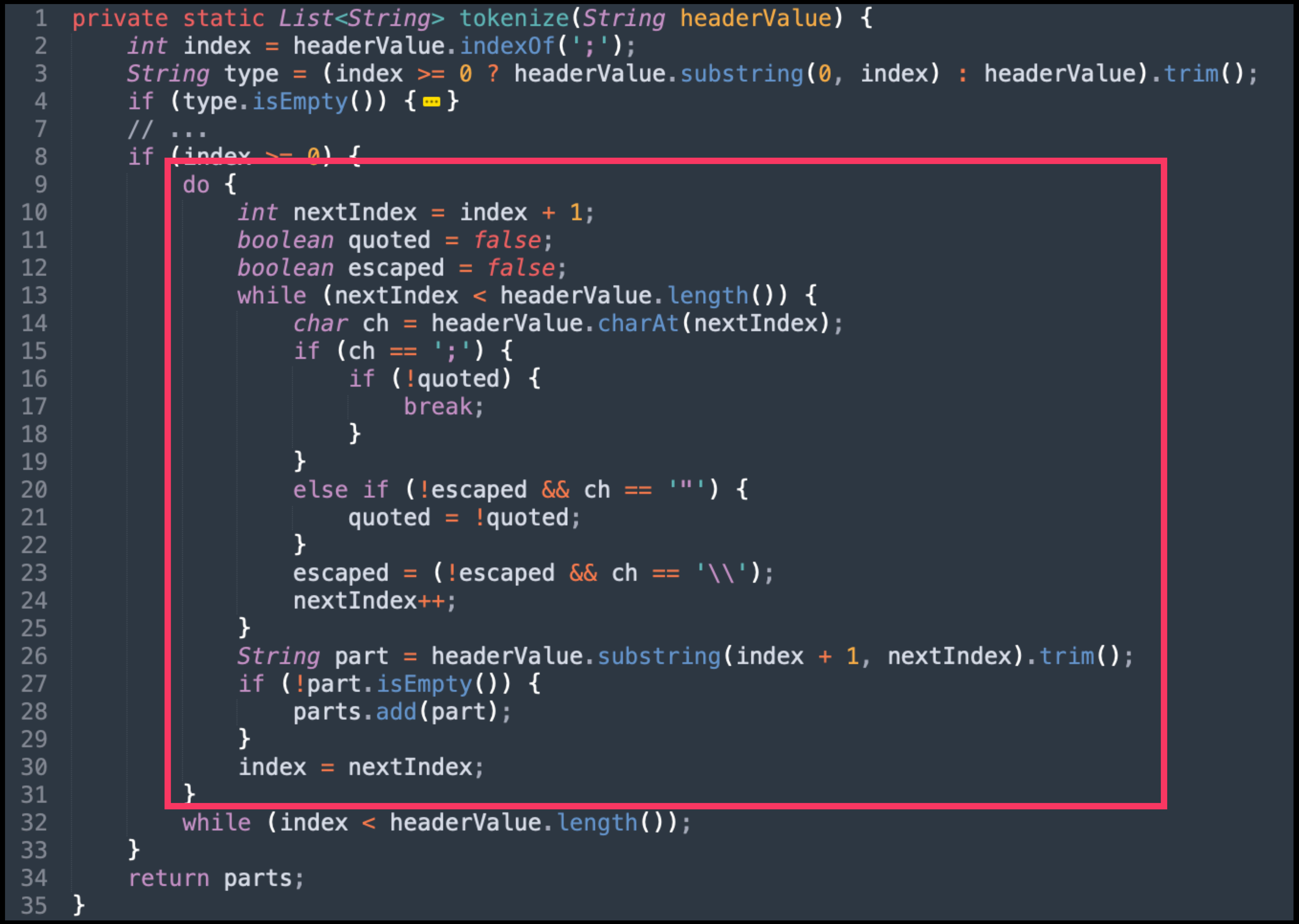
```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {⋯}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name="profile"; ↩
filename="example";';.jpg

Read Server Error - part 2



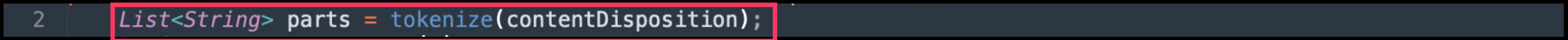
```java
List<String> parts = tokenize(contentDisposition);
```



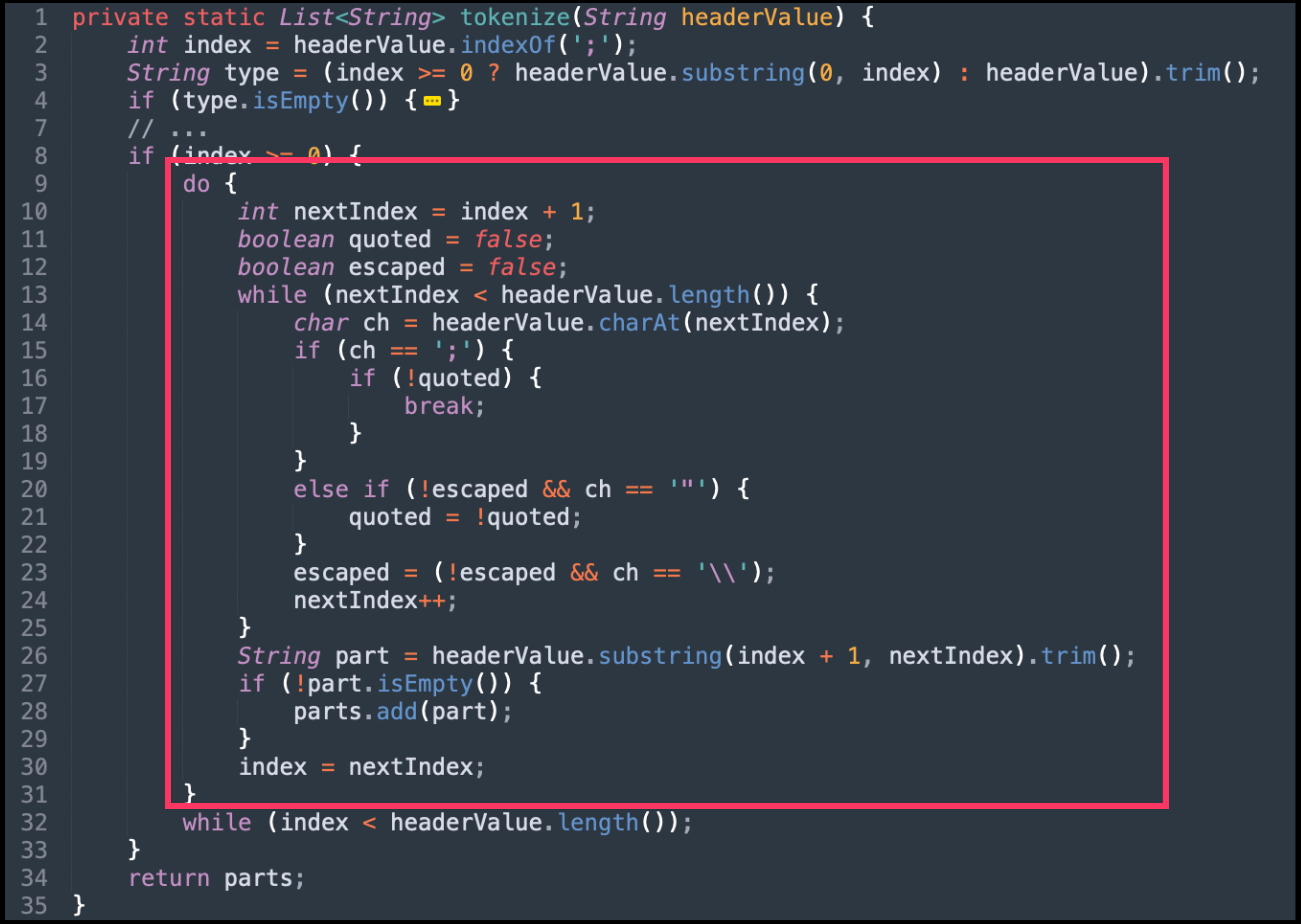
```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {⋯}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name=“profile”; ↩
filename=“example”;’;.jpg

Read Server Error - part 2



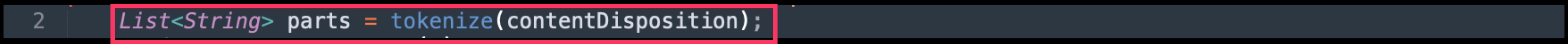
```java
List<String> parts = tokenize(contentDisposition);
```



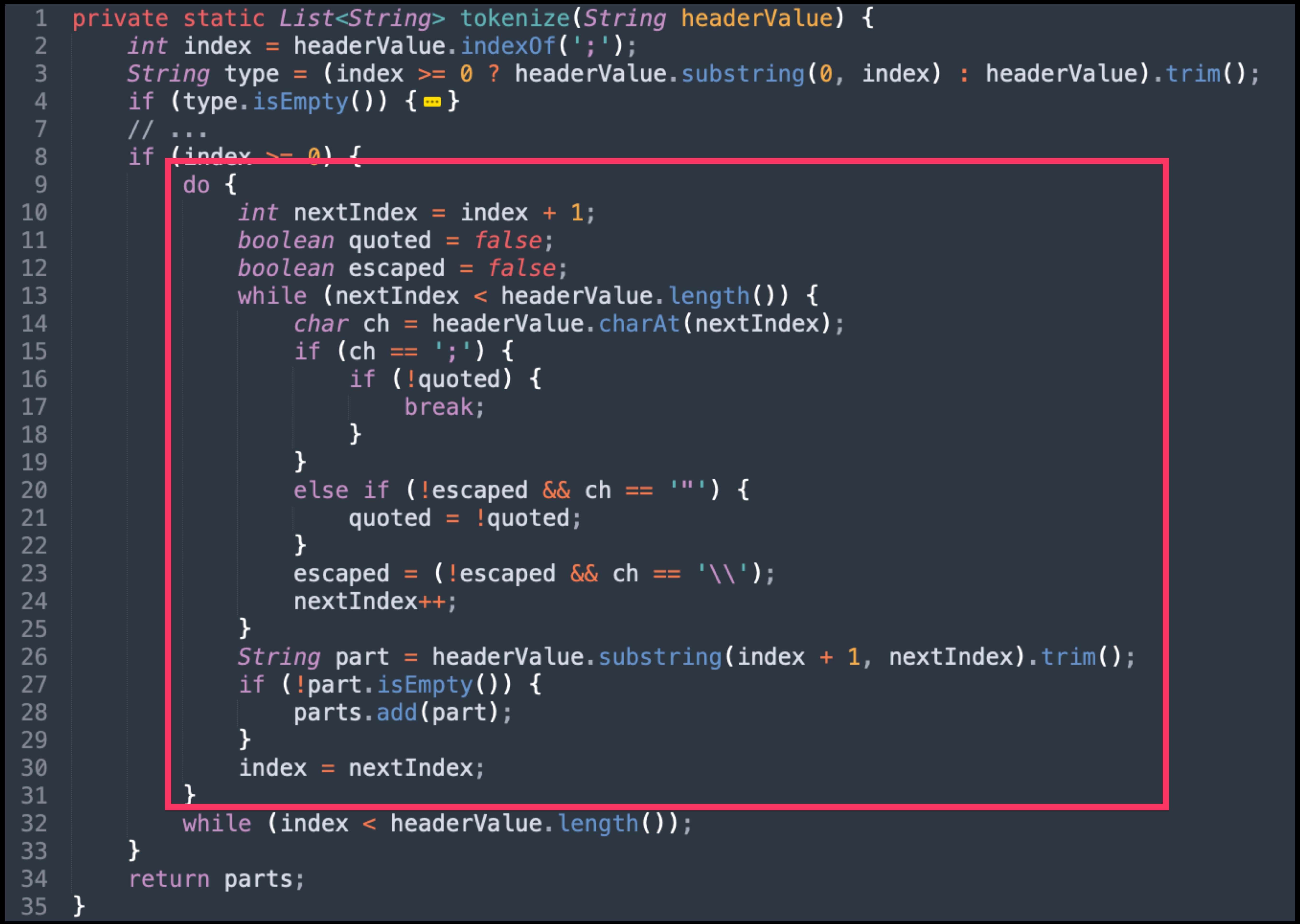
```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {▭}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name="profile"; ↩
filename="example" ; ' ; .jpg

Read Server Error - part 2



```java
List<String> parts = tokenize(contentDisposition);
```

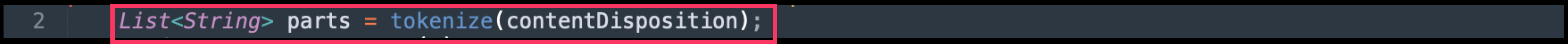


```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {⋯}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

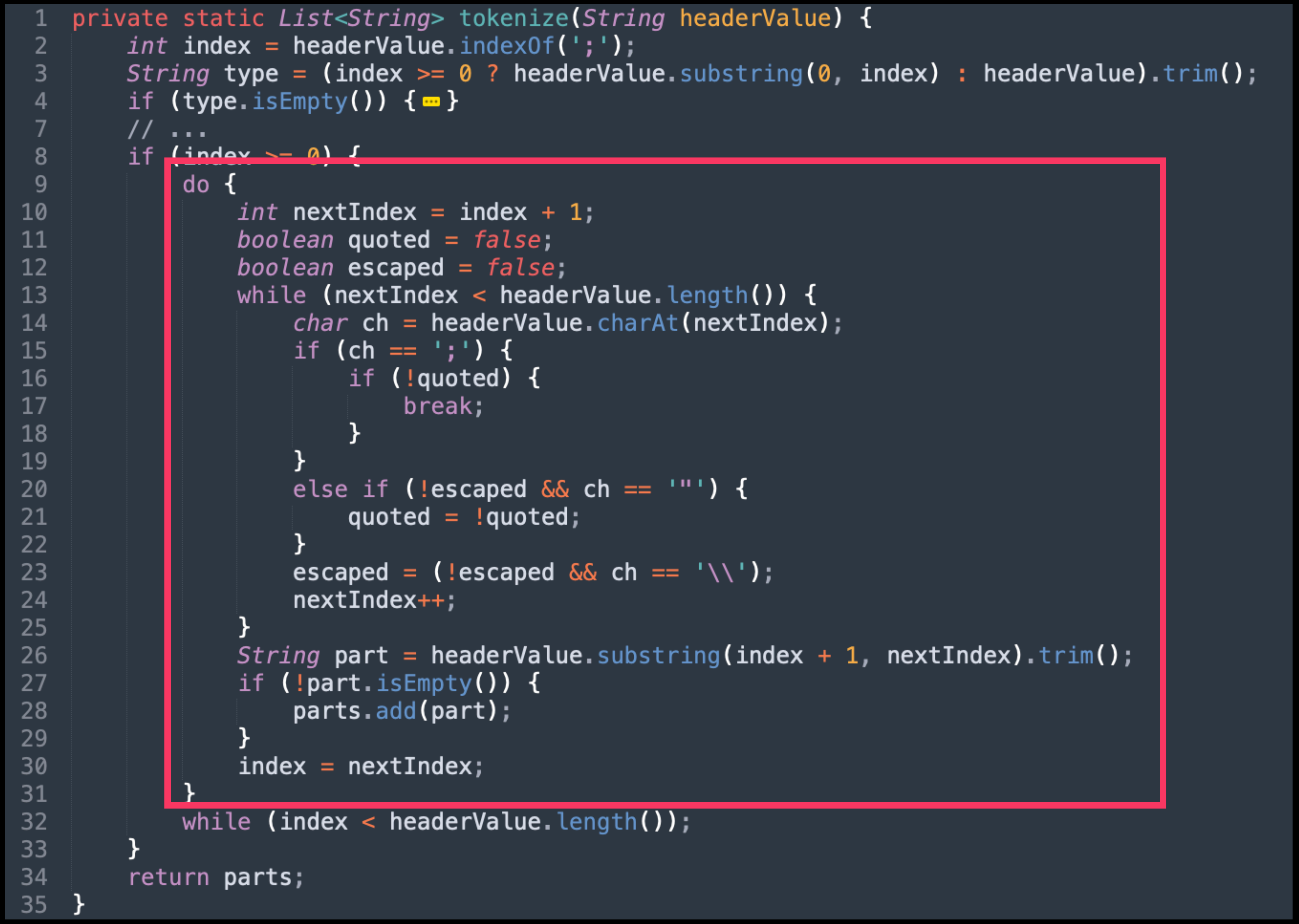
Content-Disposition: form-data; ↩
name=“profile”; ↩
filename=“example”;’;.jpg

[0]: Content-Disposition: form-data
[1]: name=“profile”
[2] filename=“example”
[3]: ’
[4]: .jpg

Read Server Error - part 2



```java
List<String> parts = tokenize(contentDisposition);
```



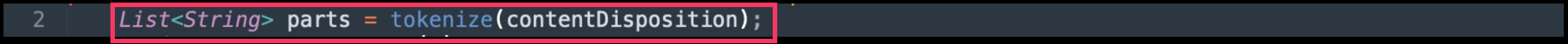
```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {…}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

Content-Disposition: form-data; ↩
name="profile"; ↩
filename="example";';.jpg

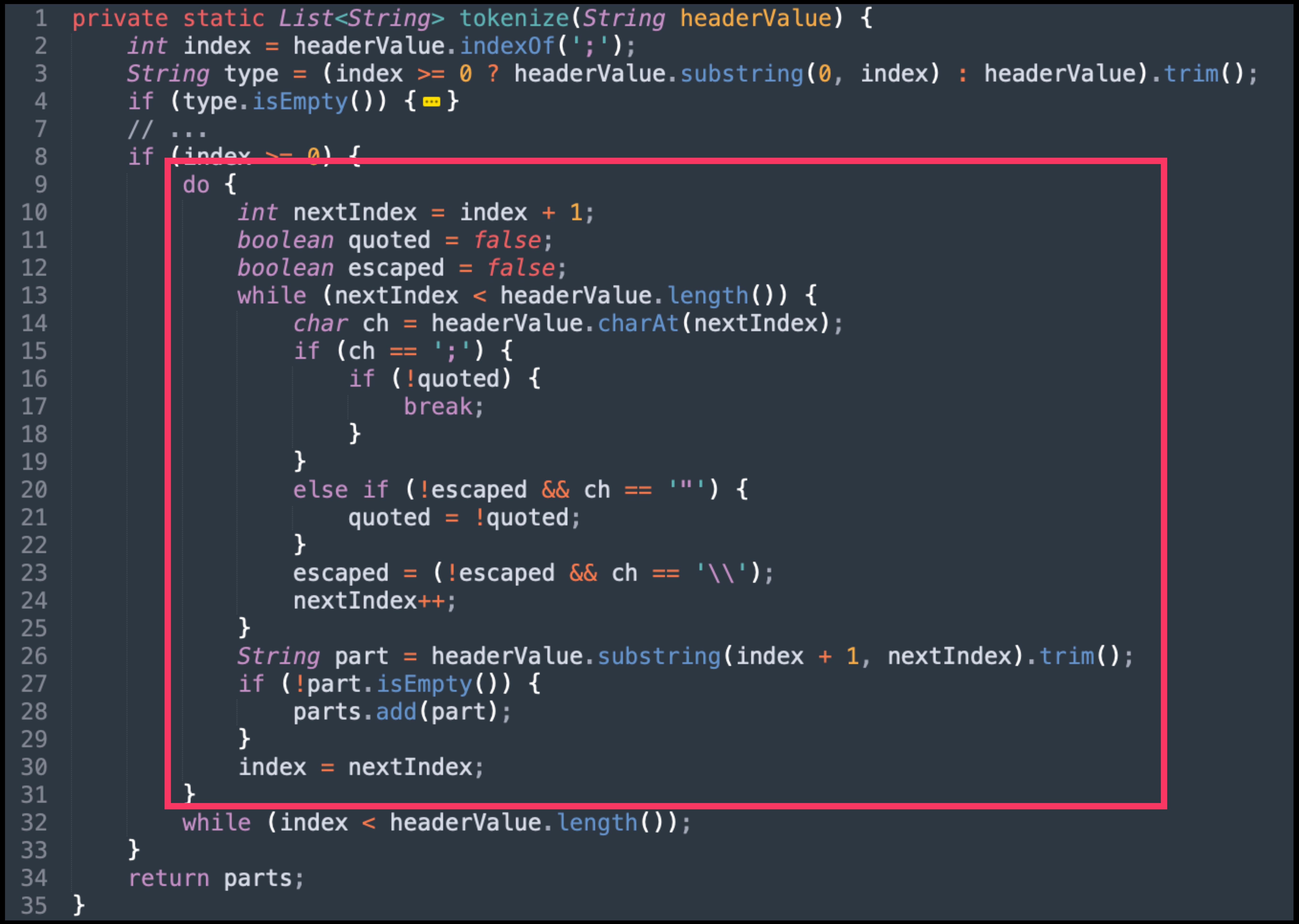
↓

[0]: Content-Disposition: form-data
[1]: name="profile"
[2] filename="example"
[3]: '
[4]: .jpg

Read Server Error - part 2



```java
List<String> parts = tokenize(contentDisposition);
```



```java
private static List<String> tokenize(String headerValue) {
    int index = headerValue.indexOf(';');
    String type = (index >= 0 ? headerValue.substring(0, index) : headerValue).trim();
    if (type.isEmpty()) {…}
    // ...
    if (index >= 0) {
        do {
            int nextIndex = index + 1;
            boolean quoted = false;
            boolean escaped = false;
            while (nextIndex < headerValue.length()) {
                char ch = headerValue.charAt(nextIndex);
                if (ch == ';') {
                    if (!quoted) {
                        break;
                    }
                }
                else if (!escaped && ch == '"') {
                    quoted = !quoted;
                }
                escaped = (!escaped && ch == '\\');
                nextIndex++;
            }
            String part = headerValue.substring(index + 1, nextIndex).trim();
            if (!part.isEmpty()) {
                parts.add(part);
            }
            index = nextIndex;
        }
        while (index < headerValue.length());
    }
    return parts;
}
```

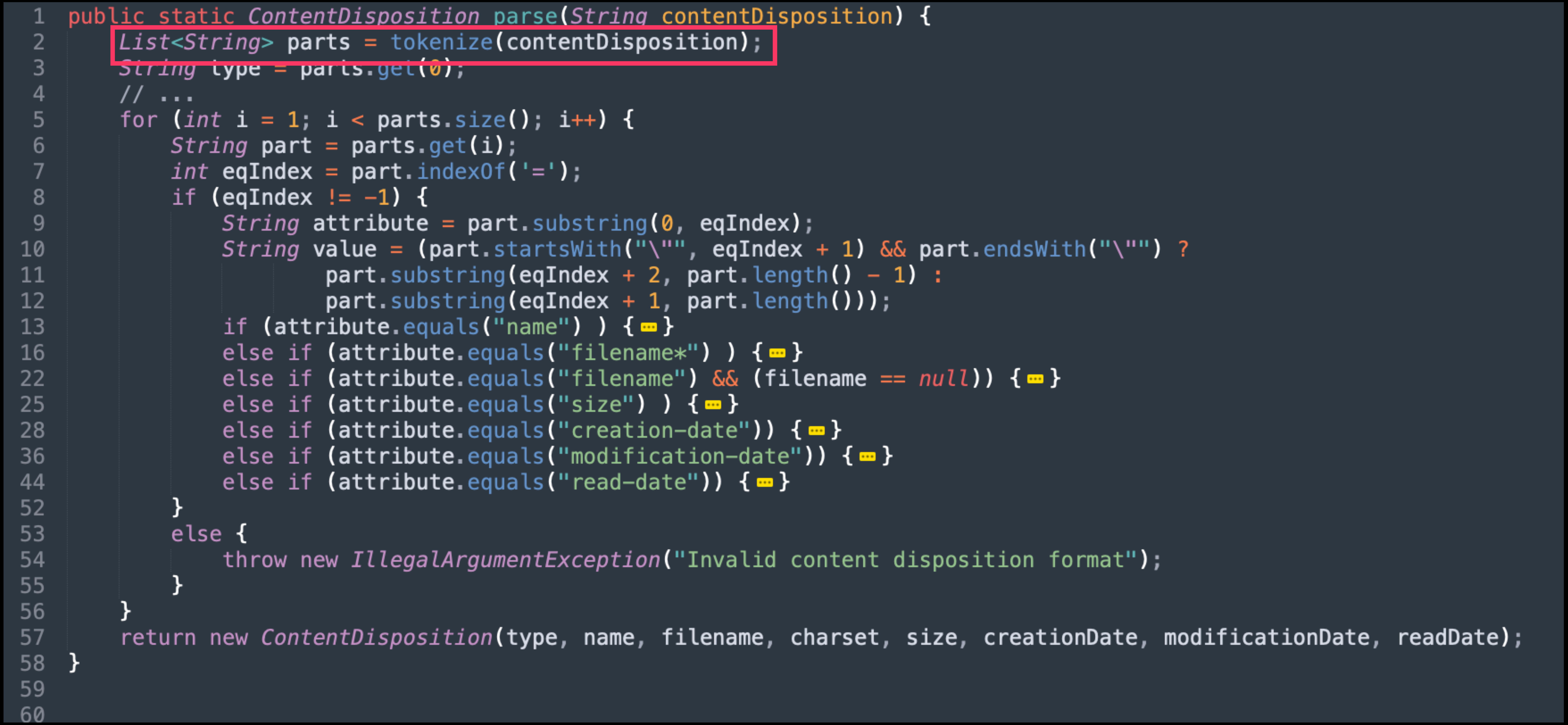
Content-Disposition: form-data; ↩
name=“profile”; ↩
filename=“example”;’;.jpg

↓

[0]: Content-Disposition: form-data
[1]: name=“profile”
[2] filename=“example”
[3]: ’
[4]: .jpg

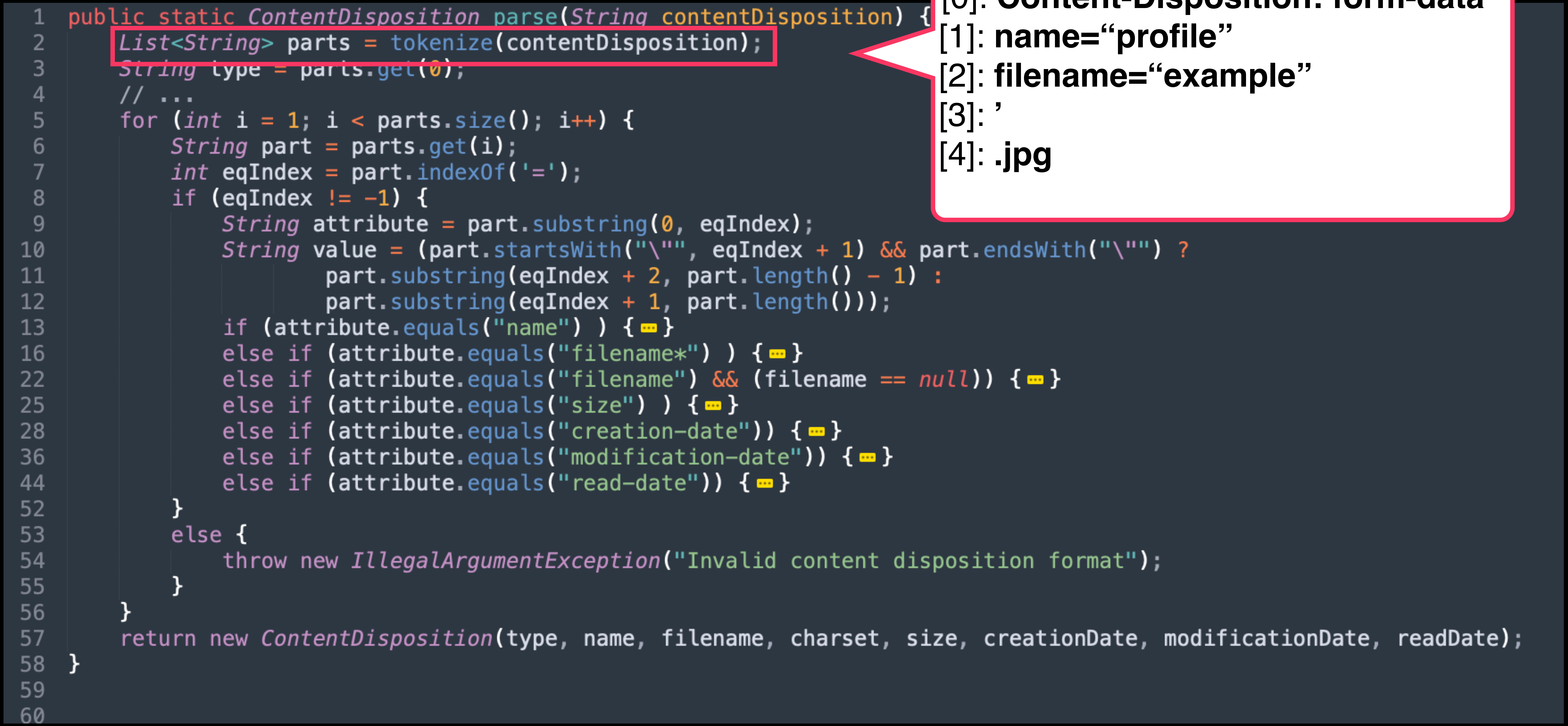
🤷‍♂️

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

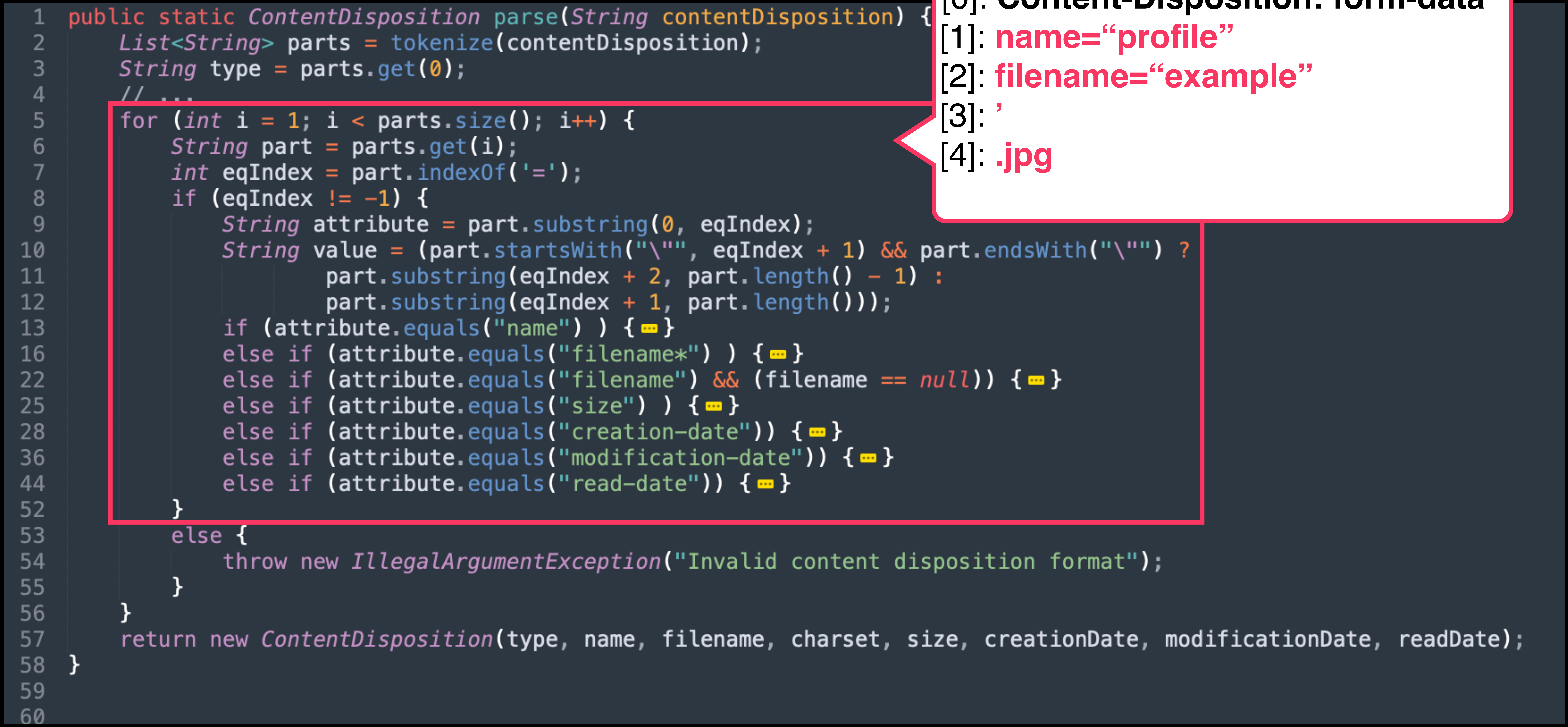
Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
  List<String> parts = tokenize(contentDisposition);
  String type = parts.get(0);
  // ...
  for (int i = 1; i < parts.size(); i++) {
    String part = parts.get(i);
    int eqIndex = part.indexOf('=');
    if (eqIndex != -1) {
      String attribute = part.substring(0, eqIndex);
      String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
          part.substring(eqIndex + 2, part.length() - 1) :
          part.substring(eqIndex + 1, part.length()));
      if (attribute.equals("name") ) {⋯}
      else if (attribute.equals("filename*") ) {⋯}
      else if (attribute.equals("filename") && (filename == null)) {⋯}
      else if (attribute.equals("size") ) {⋯}
      else if (attribute.equals("creation-date")) {⋯}
      else if (attribute.equals("modification-date")) {⋯}
      else if (attribute.equals("read-date")) {⋯}
    }
    else {
      throw new IllegalArgumentException("Invalid content disposition format");
    }
  }
  return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name=“profile”**
[2]: **filename=“example”**
[3]: **’**
[4]: **.jpg**

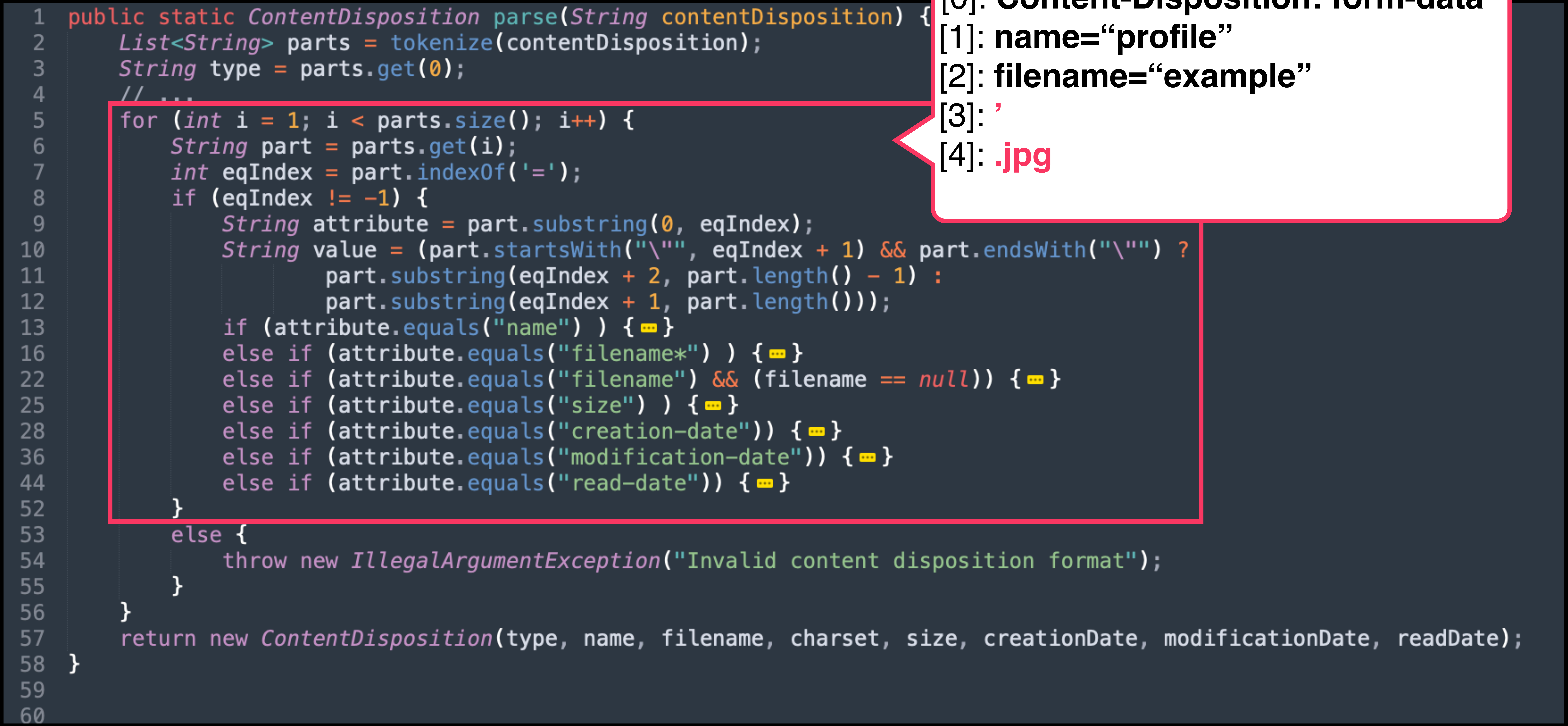
Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
  List<String> parts = tokenize(contentDisposition);
  String type = parts.get(0);
  // ...
  for (int i = 1; i < parts.size(); i++) {
    String part = parts.get(i);
    int eqIndex = part.indexOf('=');
    if (eqIndex != -1) {
      String attribute = part.substring(0, eqIndex);
      String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
          part.substring(eqIndex + 2, part.length() - 1) :
          part.substring(eqIndex + 1, part.length()));
      if (attribute.equals("name") ) {⋯}
      else if (attribute.equals("filename*") ) {⋯}
      else if (attribute.equals("filename") && (filename == null)) {⋯}
      else if (attribute.equals("size") ) {⋯}
      else if (attribute.equals("creation-date")) {⋯}
      else if (attribute.equals("modification-date")) {⋯}
      else if (attribute.equals("read-date")) {⋯}
    }
    else {
      throw new IllegalArgumentException("Invalid content disposition format");
    }
  }
  return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="profile"**
[2]: **filename="example"**
[3]: **’**
[4]: **.jpg**

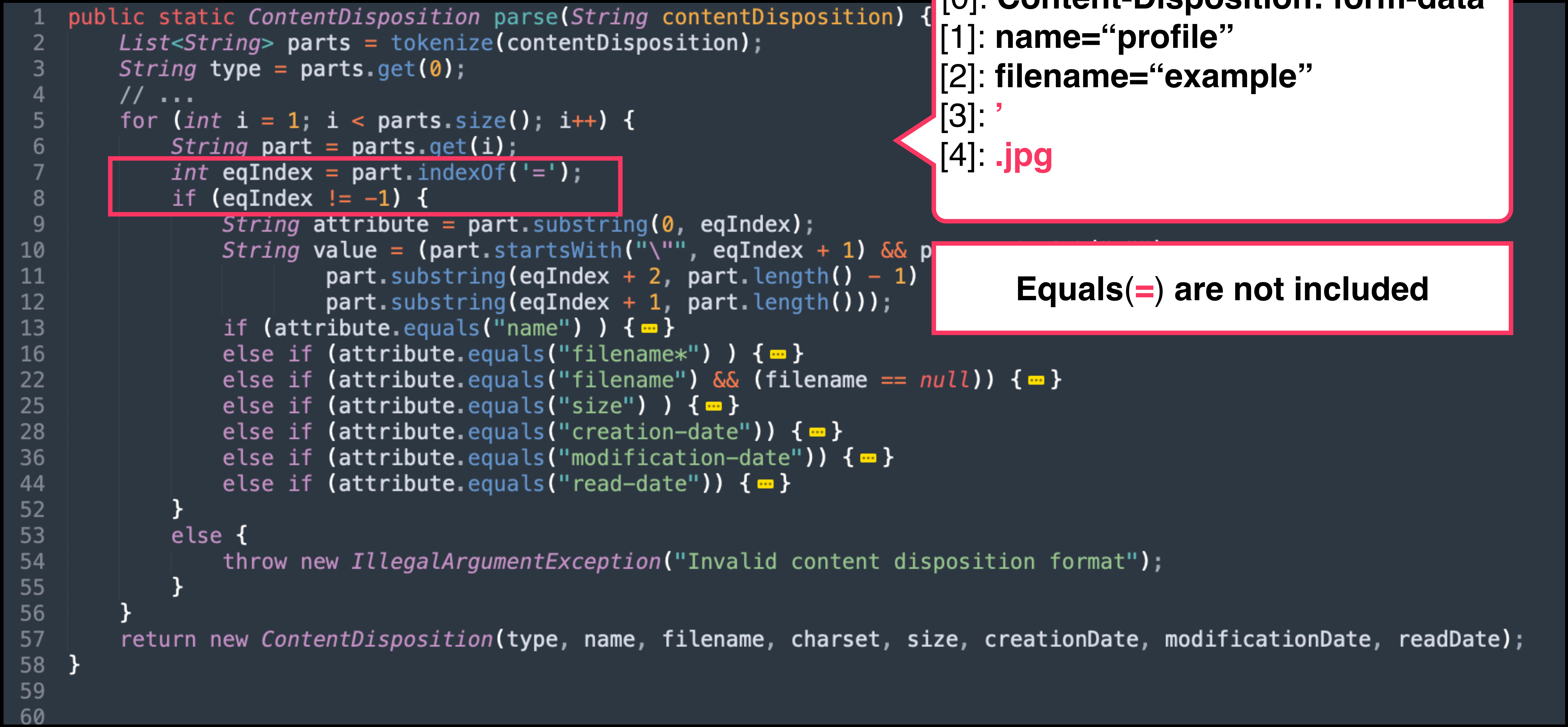
Read Server Error - part 2



```
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="profile"**
[2]: **filename="example"**
[3]: **'**
[4]: **.jpg**

Read Server Error - part 2

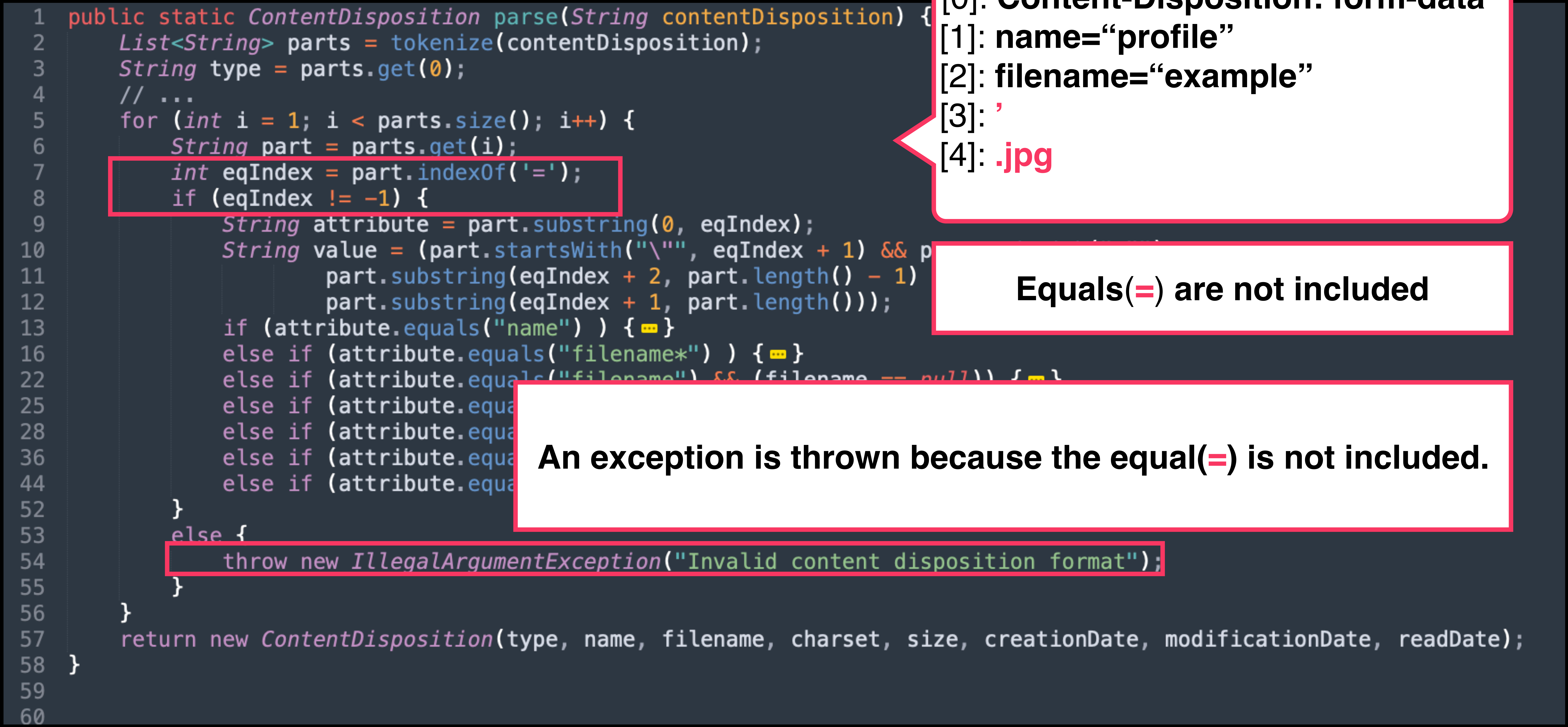


```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && p
                    part.substring(eqIndex + 2, part.length() - 1)
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="profile"**
[2]: **filename="example"**
[3]: **’**
[4]: **.jpg**

Equals(=) are not included

Read Server Error - part 2



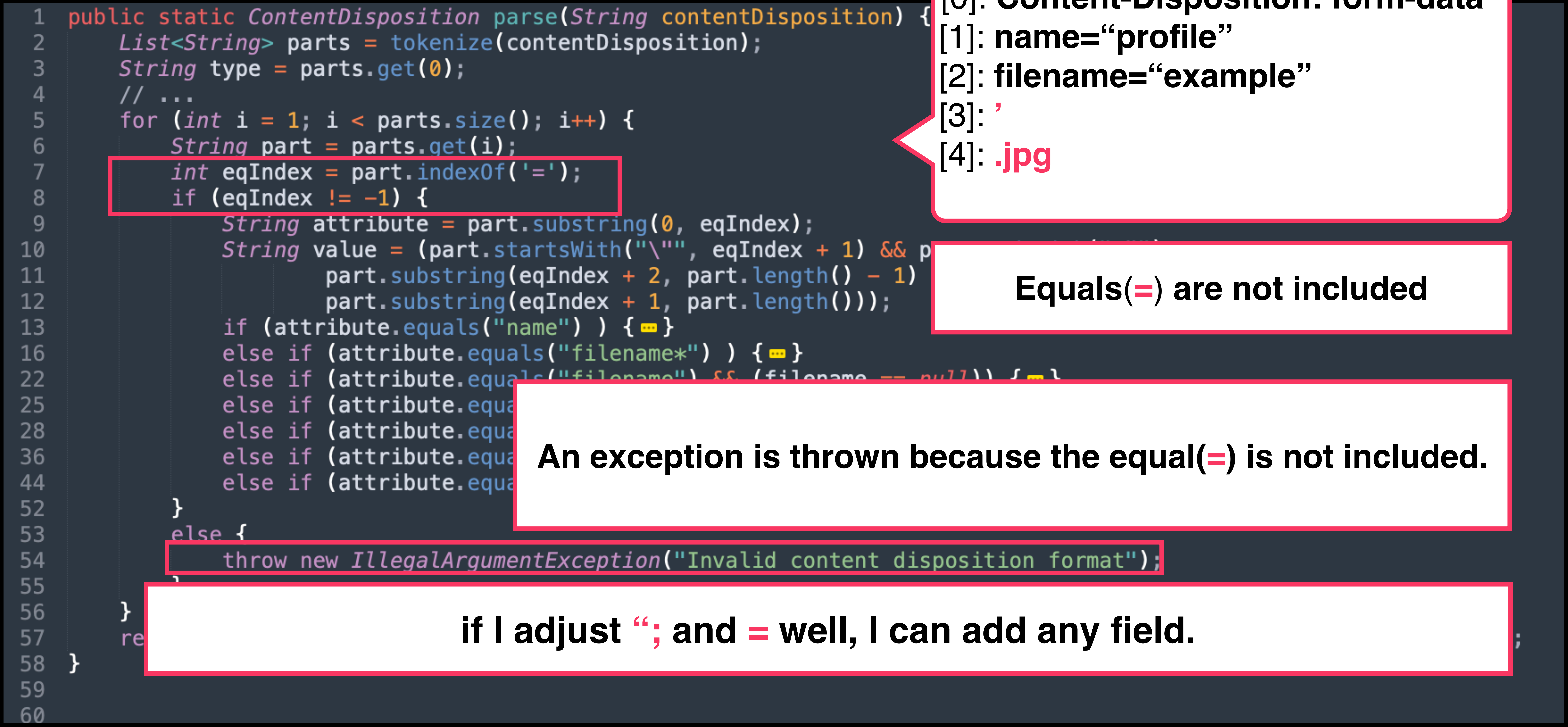
```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && p
                    part.substring(eqIndex + 2, part.length() - 1)
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {…}
            else if (attribute.equals("filename*") ) {…}
            else if (attribute.equals("filename") && (filename == null)) {…}
            else if (attribute.equa
            else if (attribute.equa
            else if (attribute.equa
            else if (attribute.equa
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    return new ContentDisposition(type, name, filename, charset, size, creationDate, modificationDate, readDate);
}
```

[0]: **Content-Disposition: form-data**
[1]: **name="profile"**
[2]: **filename="example"**
[3]: **'**
[4]: **.jpg**

Equals(=) are not included

An exception is thrown because the equal(=) is not included.

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && p
                    part.substring(eqIndex + 2, part.length() - 1)
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {...}
            else if (attribute.equals("filename*") ) {...}
            else if (attribute.equals("filename") && (filename == null)) {...}
            else if (attribute.equa
            else if (attribute.equa
            else if (attribute.equa
            else if (attribute.equa
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    re                                                                        ;
}
```

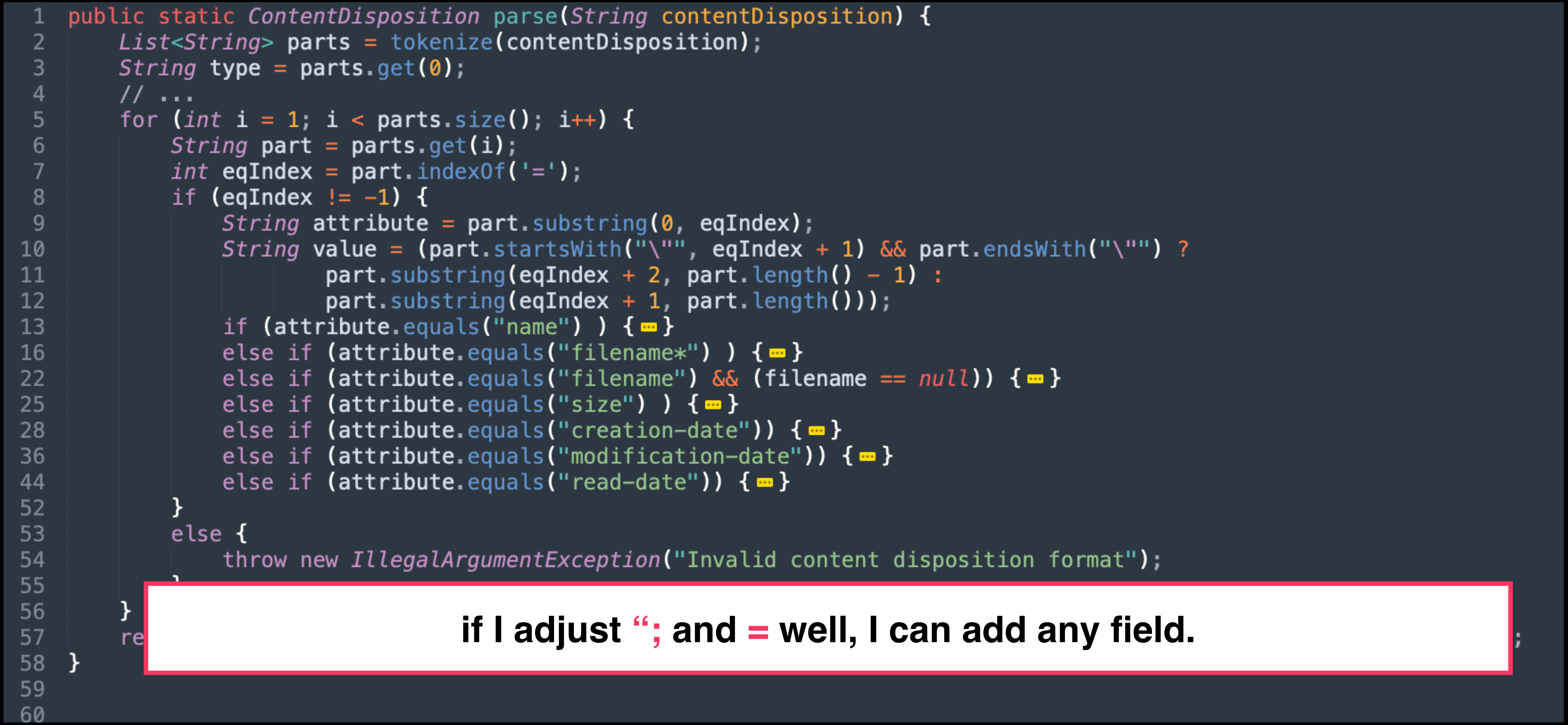
[0]: Content-Disposition: form-data
[1]: name="profile"
[2]: filename="example"
[3]: '
[4]: .jpg

Equals(=) are not included

An exception is thrown because the equal(=) is not included.

if I adjust "; and = well, I can add any field.

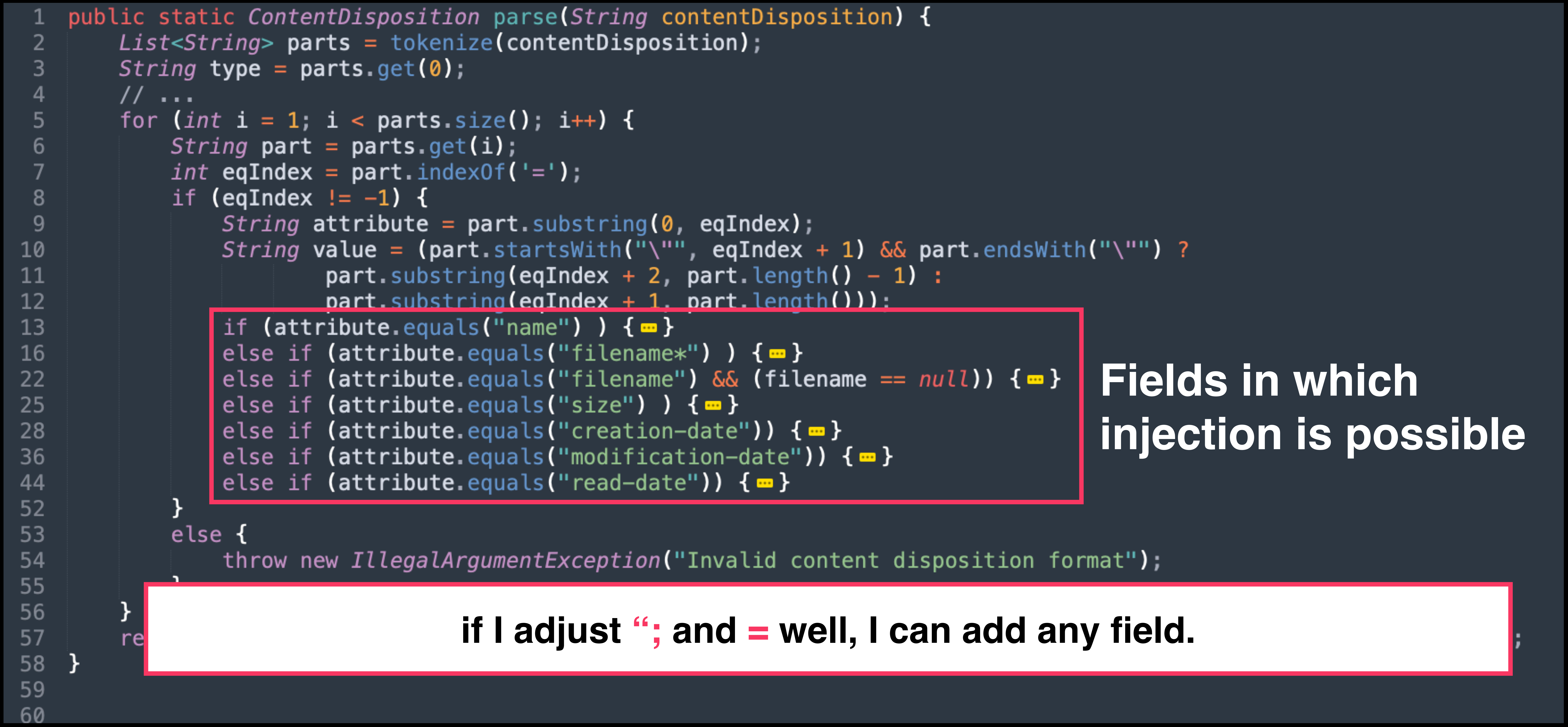
Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
  List<String> parts = tokenize(contentDisposition);
  String type = parts.get(0);
  // ...
  for (int i = 1; i < parts.size(); i++) {
    String part = parts.get(i);
    int eqIndex = part.indexOf('=');
    if (eqIndex != -1) {
      String attribute = part.substring(0, eqIndex);
      String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
          part.substring(eqIndex + 2, part.length() - 1) :
          part.substring(eqIndex + 1, part.length()));
      if (attribute.equals("name") ) {⋯}
      else if (attribute.equals("filename*") ) {⋯}
      else if (attribute.equals("filename") && (filename == null)) {⋯}
      else if (attribute.equals("size") ) {⋯}
      else if (attribute.equals("creation-date")) {⋯}
      else if (attribute.equals("modification-date")) {⋯}
      else if (attribute.equals("read-date")) {⋯}
    }
    else {
      throw new IllegalArgumentException("Invalid content disposition format");
    }
  }
  re
}
```

if I adjust “; and = well, I can add any field.

Read Server Error - part 2



```java
public static ContentDisposition parse(String contentDisposition) {
    List<String> parts = tokenize(contentDisposition);
    String type = parts.get(0);
    // ...
    for (int i = 1; i < parts.size(); i++) {
        String part = parts.get(i);
        int eqIndex = part.indexOf('=');
        if (eqIndex != -1) {
            String attribute = part.substring(0, eqIndex);
            String value = (part.startsWith("\"", eqIndex + 1) && part.endsWith("\"") ?
                    part.substring(eqIndex + 2, part.length() - 1) :
                    part.substring(eqIndex + 1, part.length()));
            if (attribute.equals("name") ) {⋯}
            else if (attribute.equals("filename*") ) {⋯}
            else if (attribute.equals("filename") && (filename == null)) {⋯}
            else if (attribute.equals("size") ) {⋯}
            else if (attribute.equals("creation-date")) {⋯}
            else if (attribute.equals("modification-date")) {⋯}
            else if (attribute.equals("read-date")) {⋯}
        }
        else {
            throw new IllegalArgumentException("Invalid content disposition format");
        }
    }
    re                                                                        ;
}
```

Fields in which injection is possible

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

```
[0]: Content-Disposition: form-data
[1]: name=“profile”
[2]: filename=“example”
[3]: ’
[4]: .jpg
```

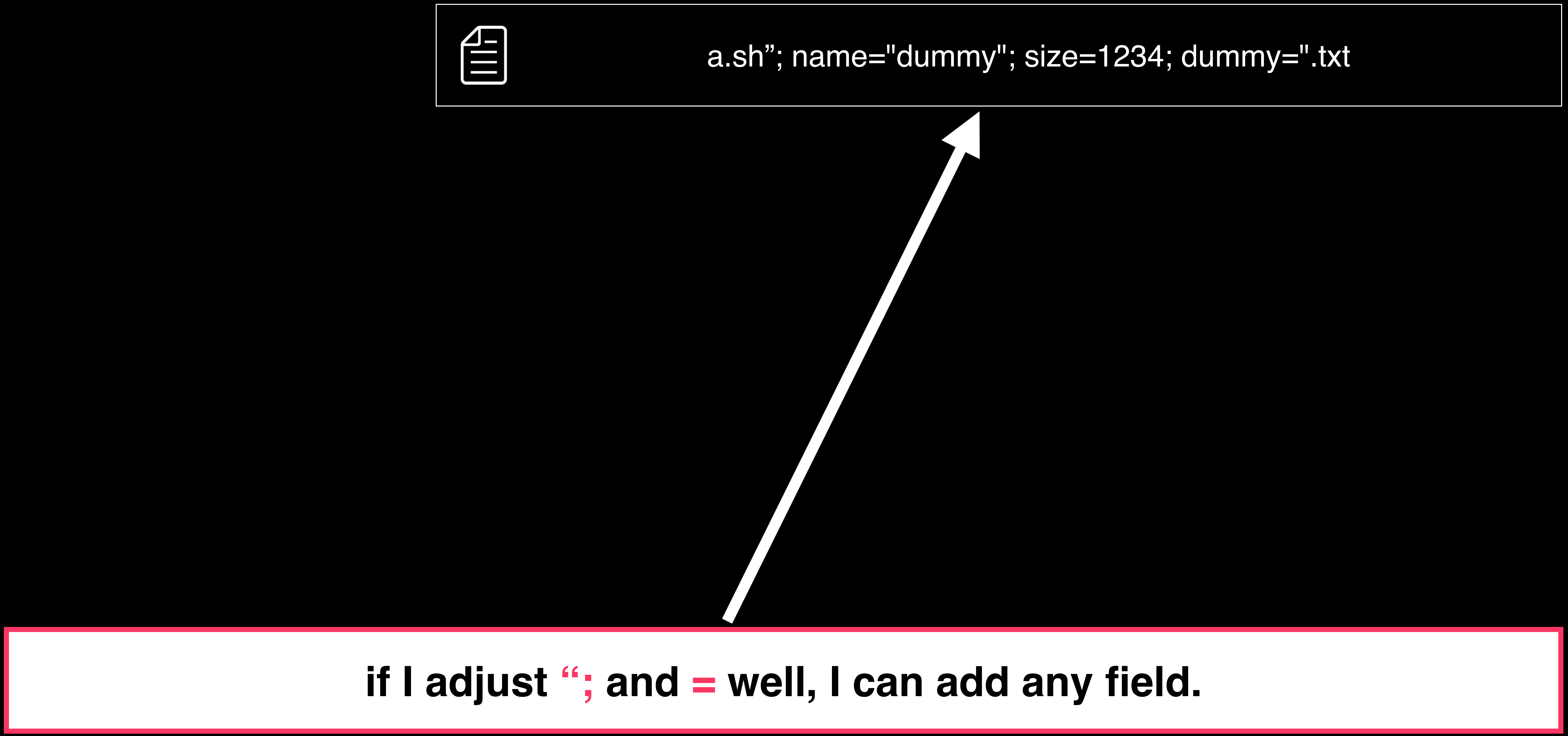
if I adjust “; and = well, I can add any field.

Read Server Error - part 2

[0]: Content-Disposition: form-data
[1]: name=“profile”
[2]: filename=“example”
[3]: x=“““’””
[4]: x=“.jpg”

if I adjust “; and = well, I can add any field.

Read Server Error - part 2



a.sh”; name="dummy"; size=1234; dummy=".txt

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

if I adjust “; and = well, I can add any field.

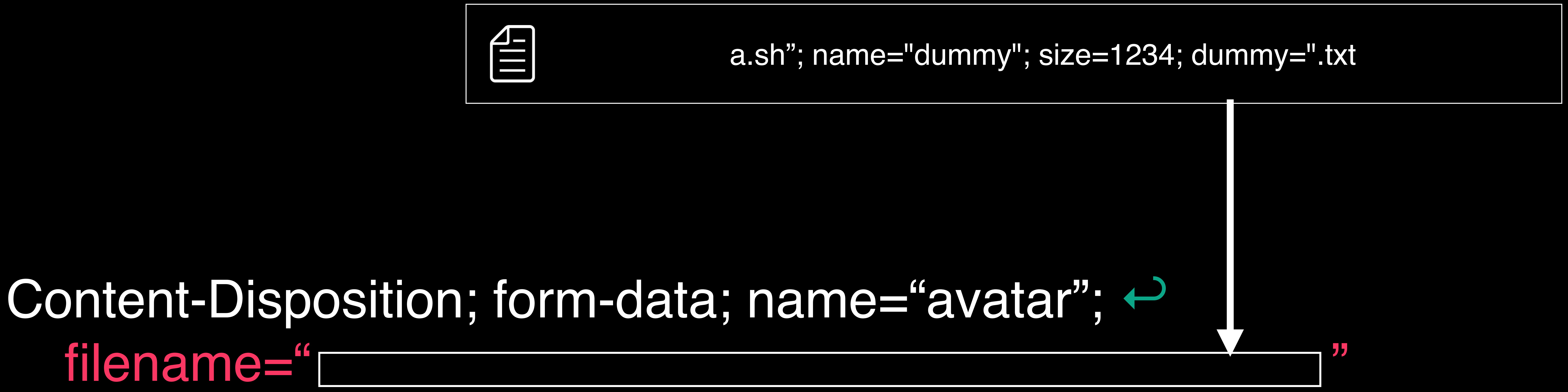
Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

Content-Disposition; form-data; name=“avatar”; ↩
filename=“ ”

if I adjust “; and = well, I can add any field.

Read Server Error - part 2



a.sh”; name="dummy"; size=1234; dummy=".txt

Content-Disposition; form-data; name=“avatar”; ↩
filename=“ ”

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

Content-Disposition; form-data; name=“avatar”; ↩
filename=“a.sh”; name="dummy"; size=1234; dummy=".txt ”

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

🗎 a.sh”; name="dummy"; size=1234; dummy=".txt

```
Content-Disposition; form-data; name=“avatar”; ↩
    filename=“a.sh”; name="dummy"; size=1234; dummy=".txt ”
```

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

name	**dummy**
filename	**a.sh**
size	**1234**
creation-date	
modification-date	
read-date	

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

Field name can be overwritten.
(e.g. “user_file” —> “dummy”)

a.sh”; name="dummy"; size=1234; dummy=".txt

name	**dummy**
filename	**a.sh**
size	**1234**
creation-date	
modification-date	
read-date	

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

Extension can be changed.
(e.g. “normal.txt” —> “malicious.sh”)

name	**dummy**
filename	**a.sh**
size	**1234**
creation-date	
modification-date	
read-date	

if I adjust “; and = well, I can add any field.

Read Server Error - part 2

a.sh”; name="dummy"; size=1234; dummy=".txt

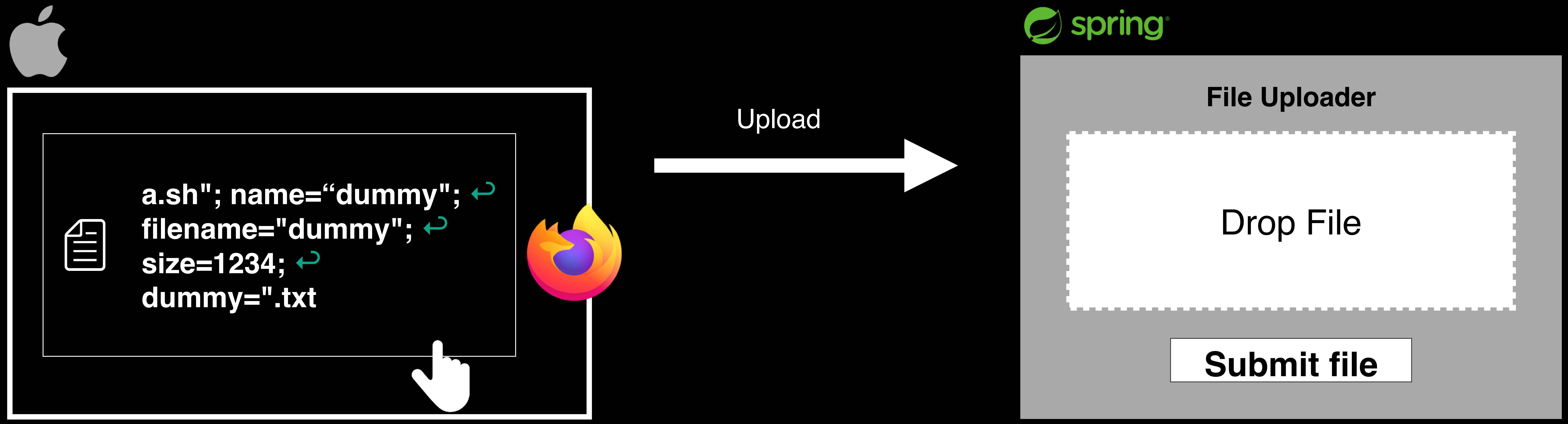
↓

name	**dummy**
filename	**a.sh**
size	**1234**
creation-date	
modification-date	
read-date	

Unused in Spring and therefore not exploitable.

if I adjust “; and = well, I can add any field.

RE: Signs of a problem



a.sh"; name="dummy"; ↩
filename="dummy"; ↩
size=1234; ↩
dummy=".txt

Upload

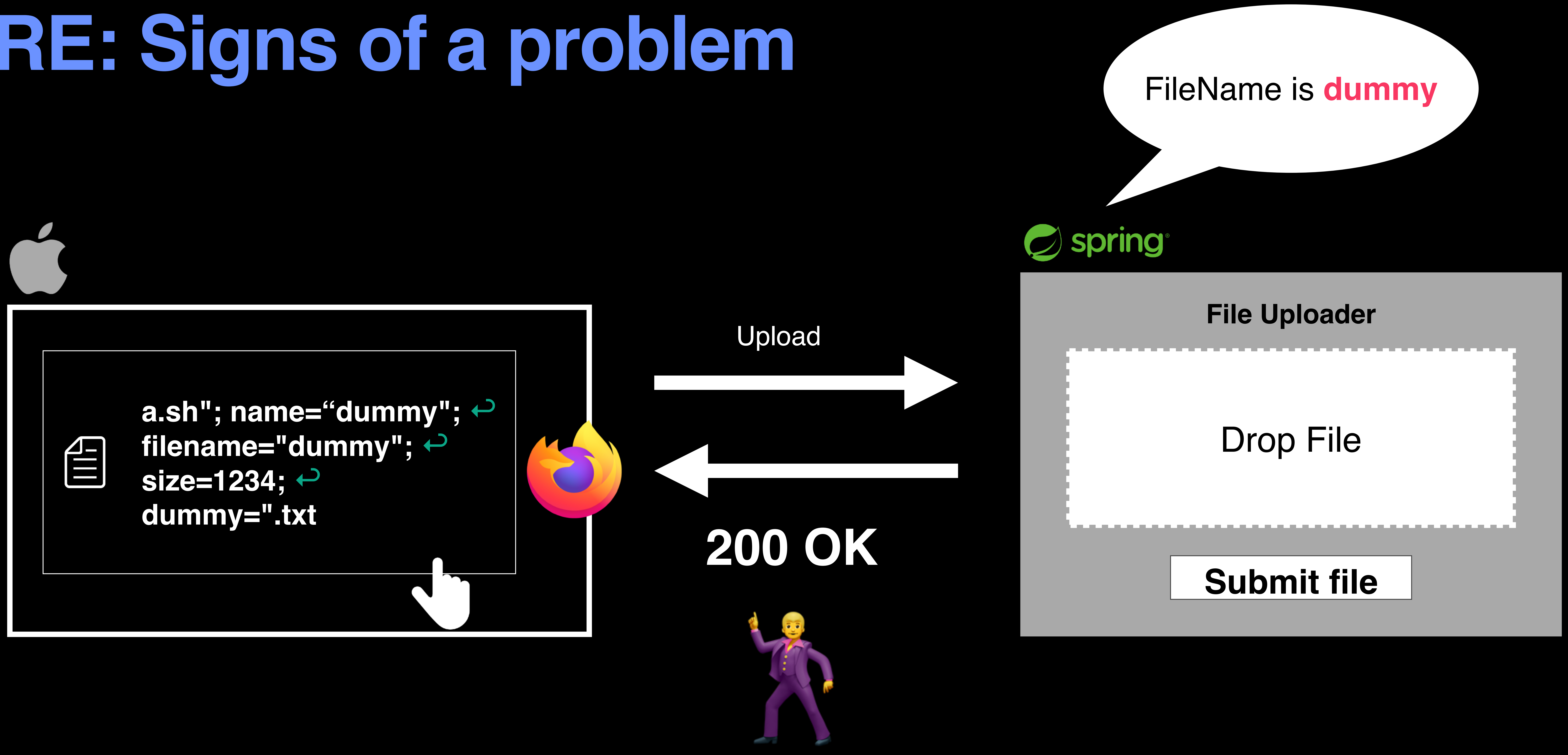
spring

File Uploader

Drop File

Submit file

RE: Signs of a problem



FileName is **dummy**

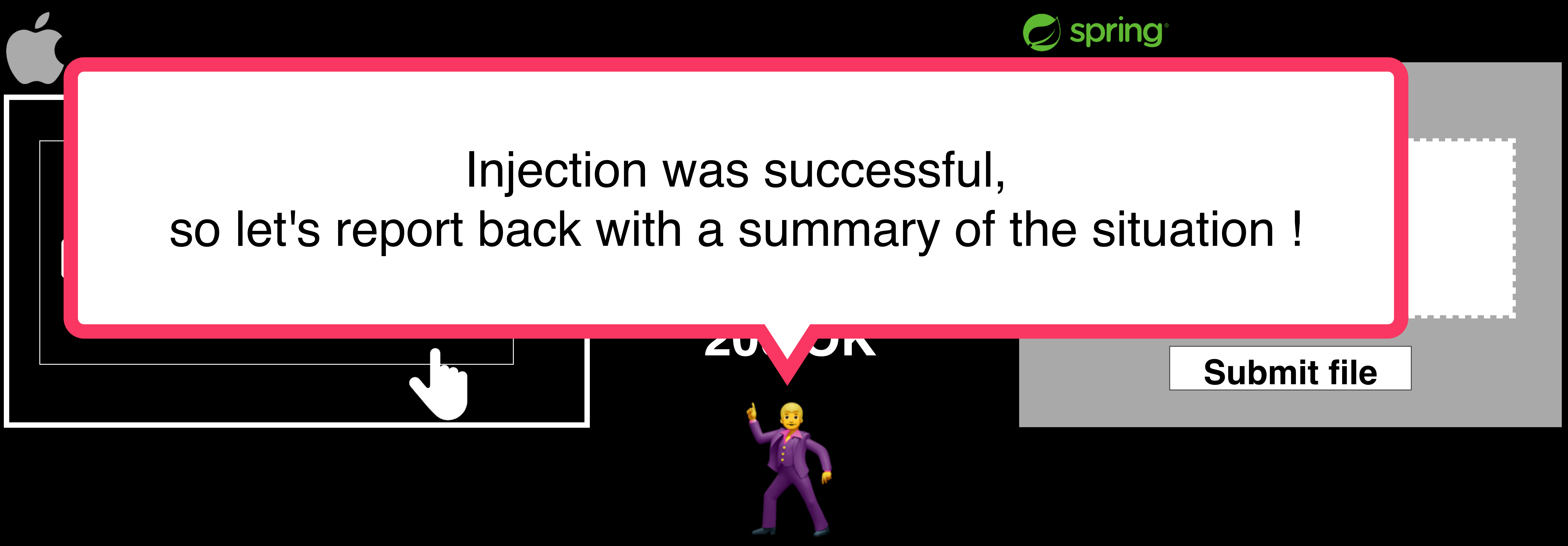
```
a.sh"; name=“dummy"; ↩
filename="dummy"; ↩
size=1234; ↩
dummy=".txt
```

Upload

200 OK

Inject Succeeded

RE: Signs of a problem

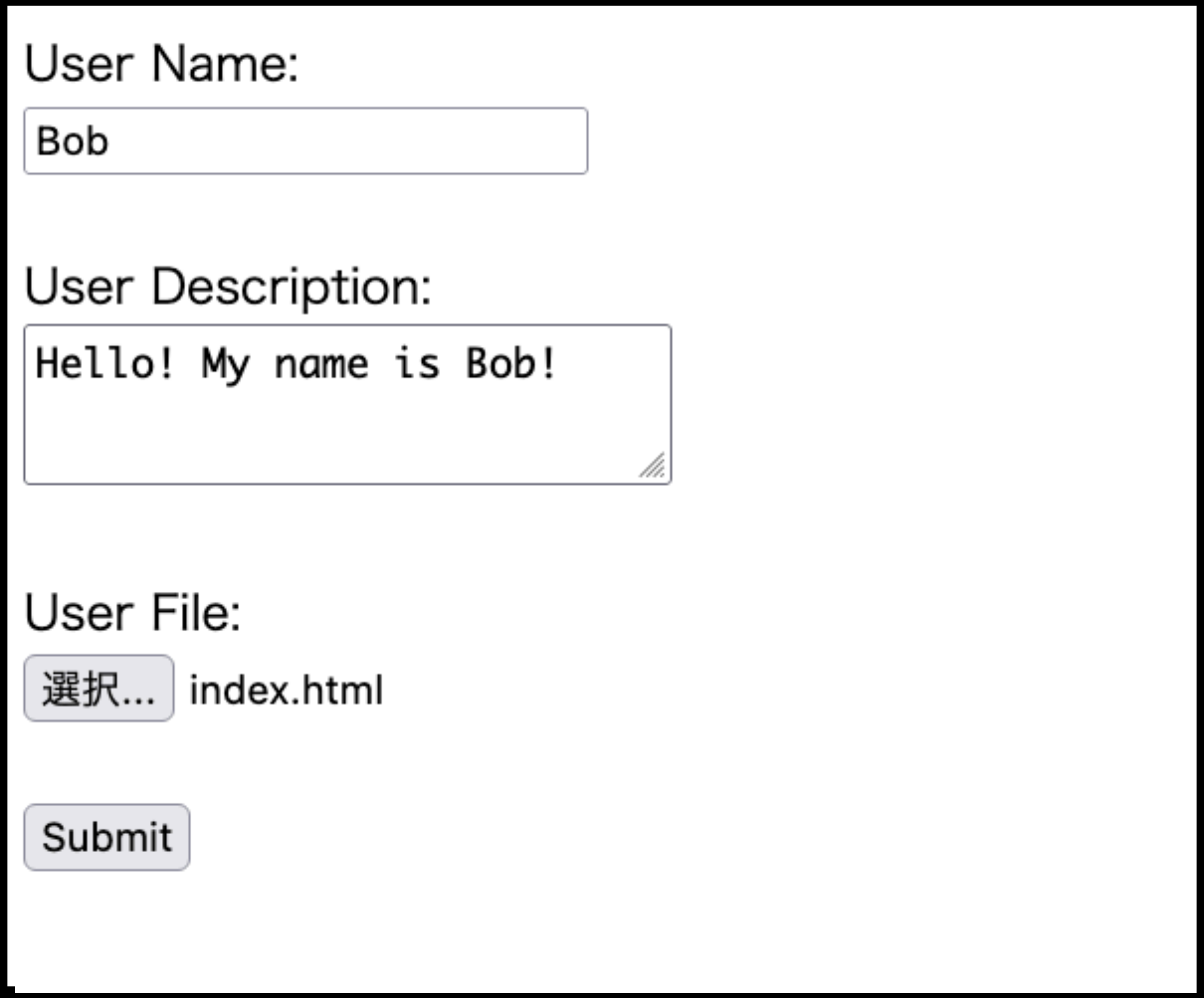


Injection was successful,
so let's report back with a summary of the situation !

Inject Succeeded

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

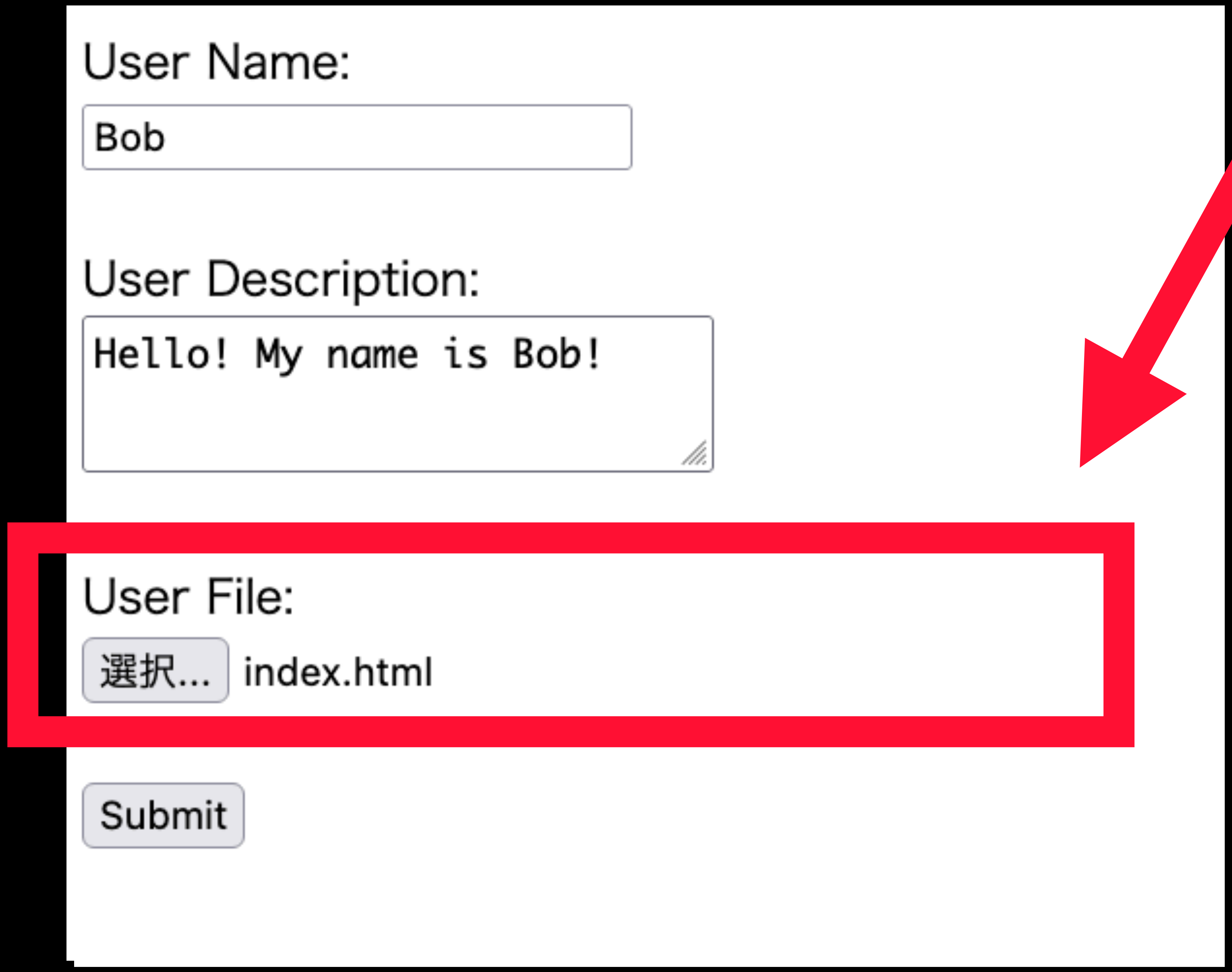
index.html

spring

user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

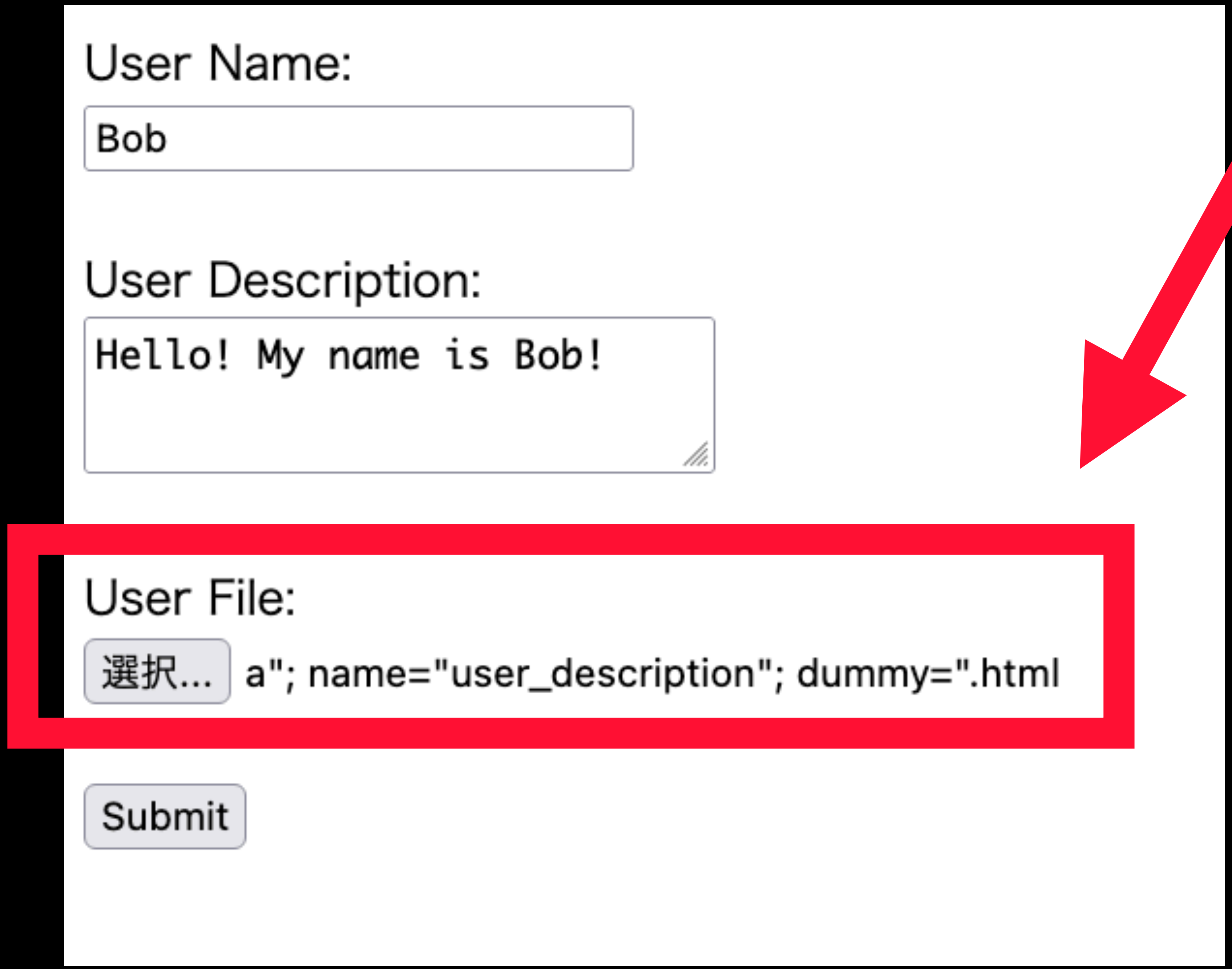
index.html

spring

user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

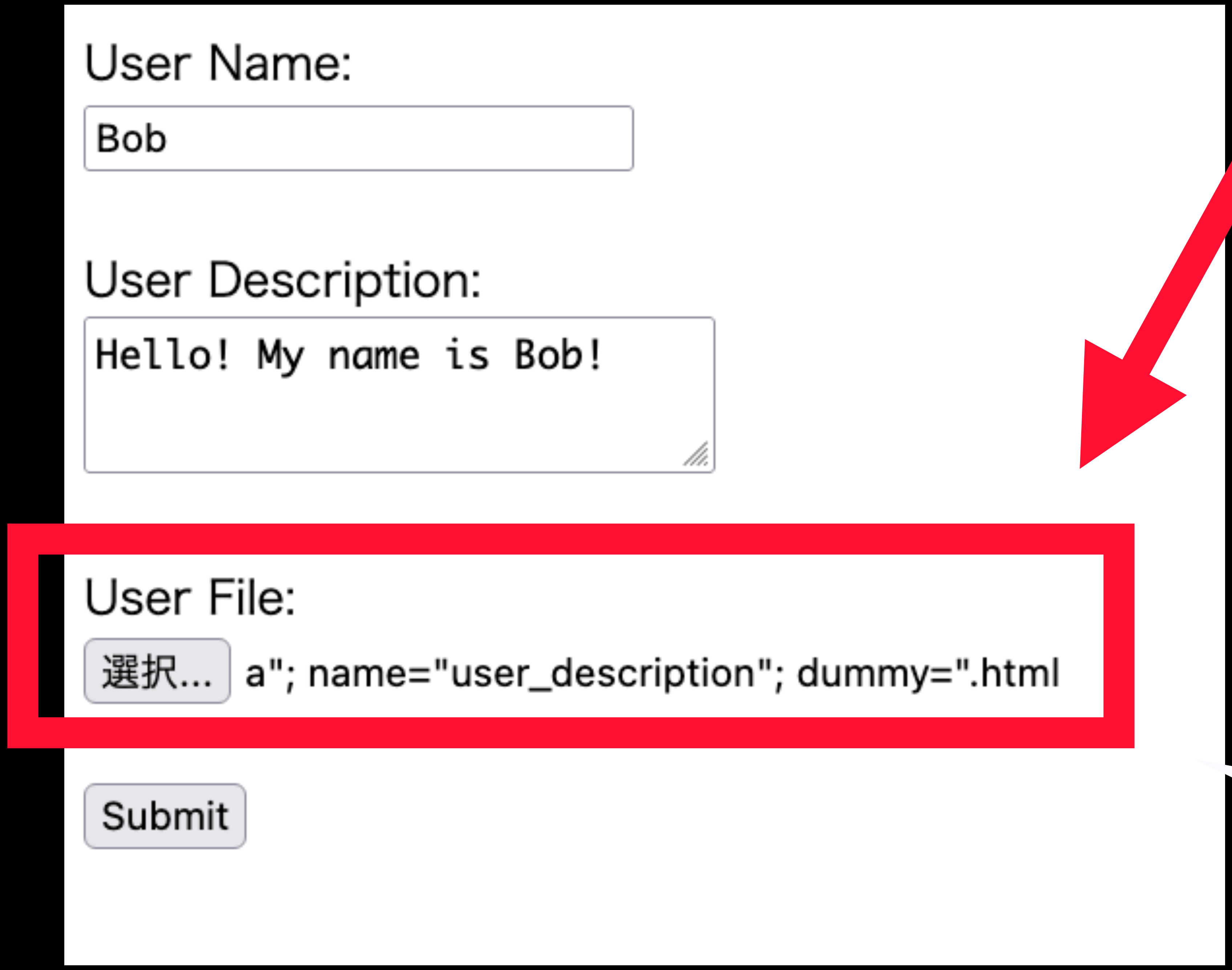
a"; name="user_description"; dummy=".html

spring

user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

a"; name="user_description"; dummy=".html

spring

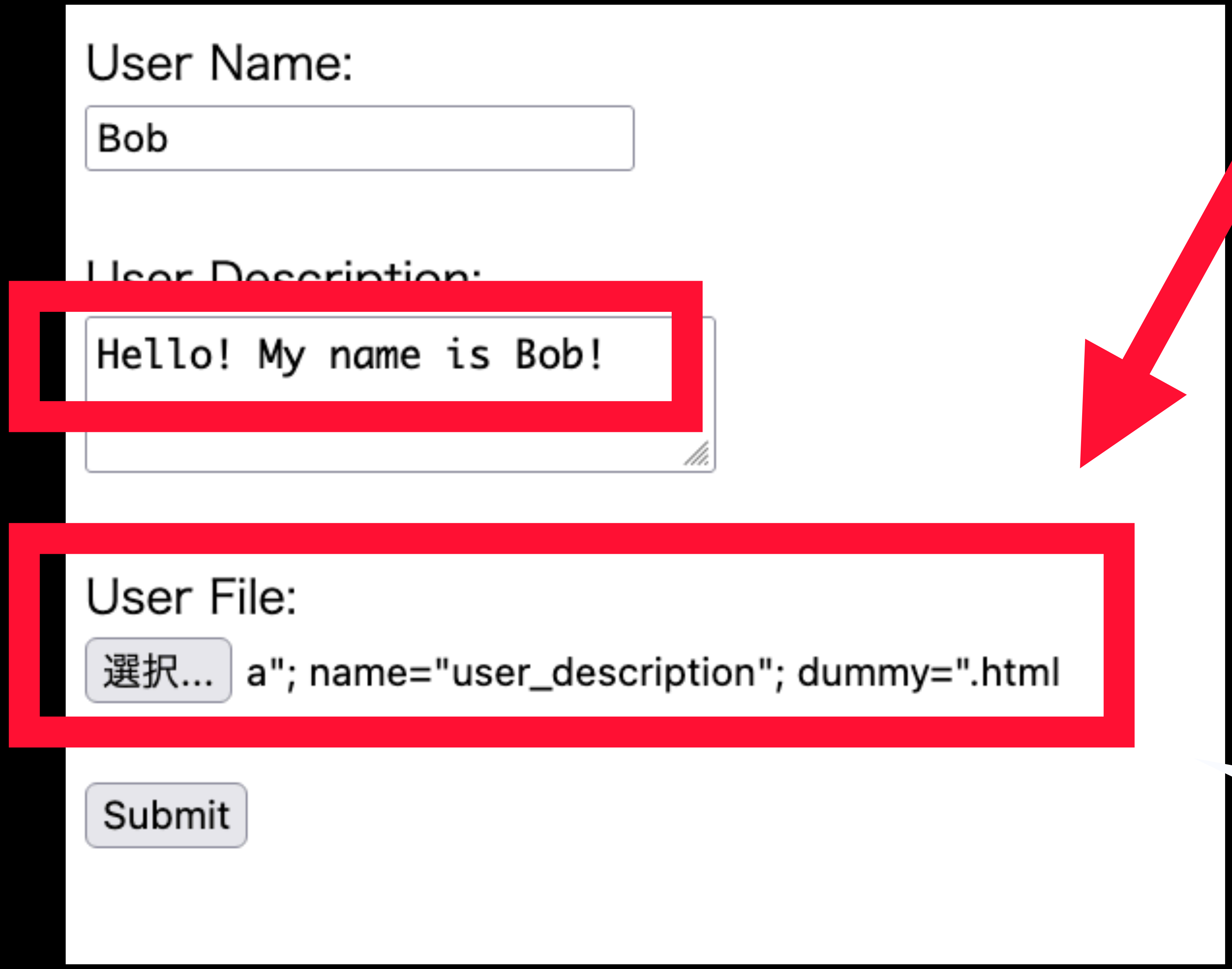
user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

(File Content)

My name is Mallory.

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

選択... a"; name="user_description"; dummy=".html

Submit

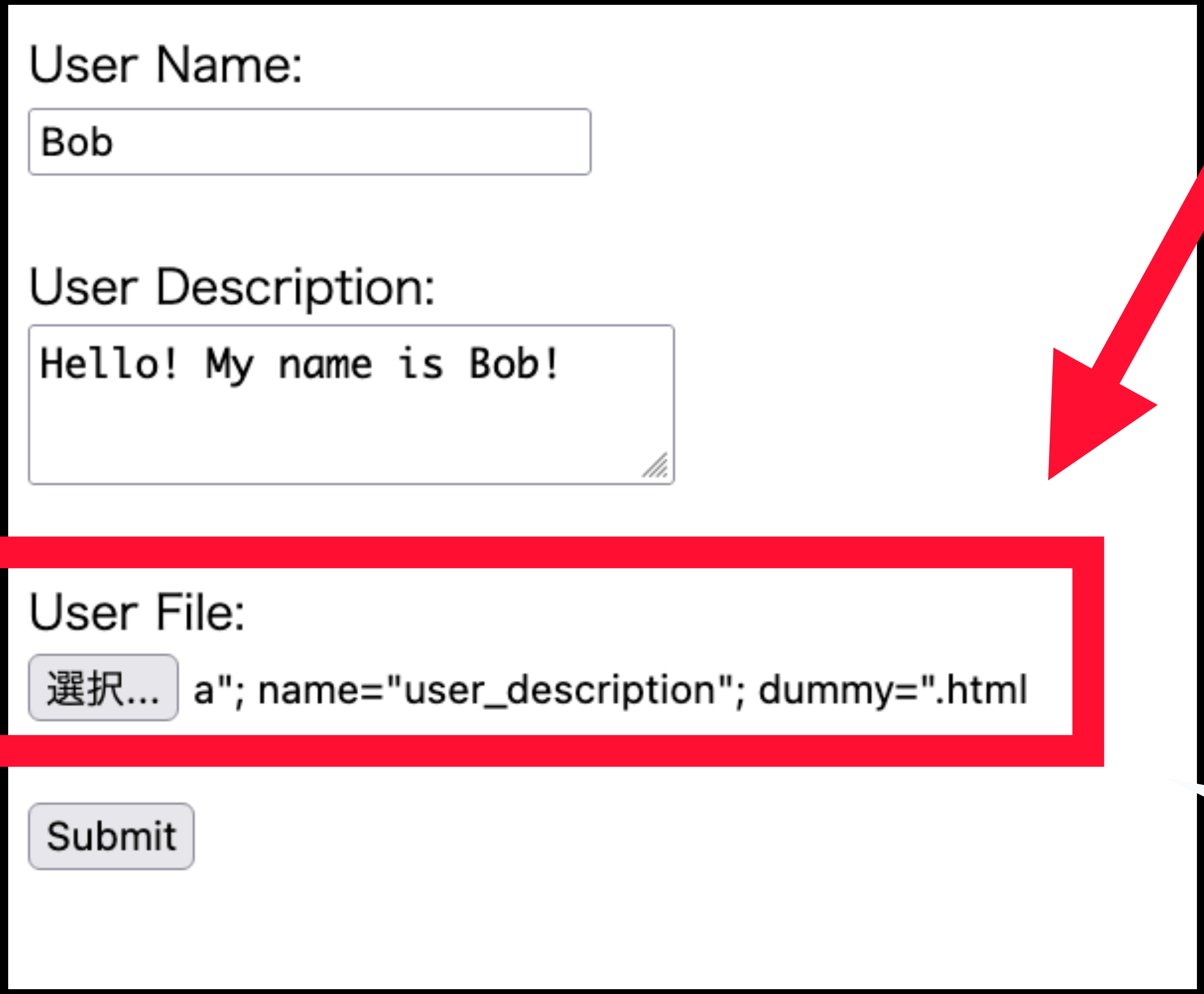
user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

(File Content)

My name is Mallory.

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

選択... a"; name="user_description"; dummy=".html

spring

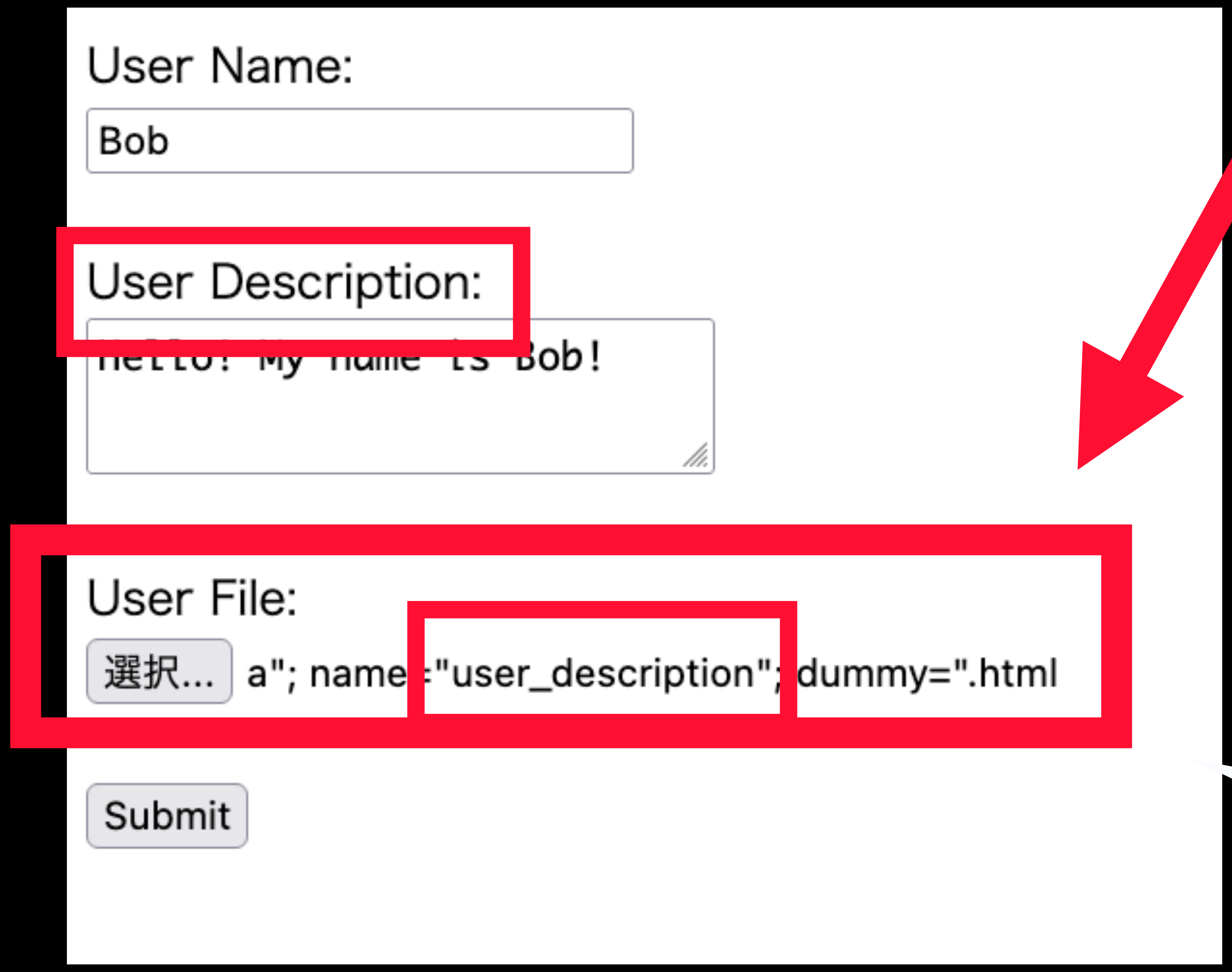
user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

(File Content)

My name is Mallory.

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

選択... a"; name="user_description"; dummy=".html

Submit

spring

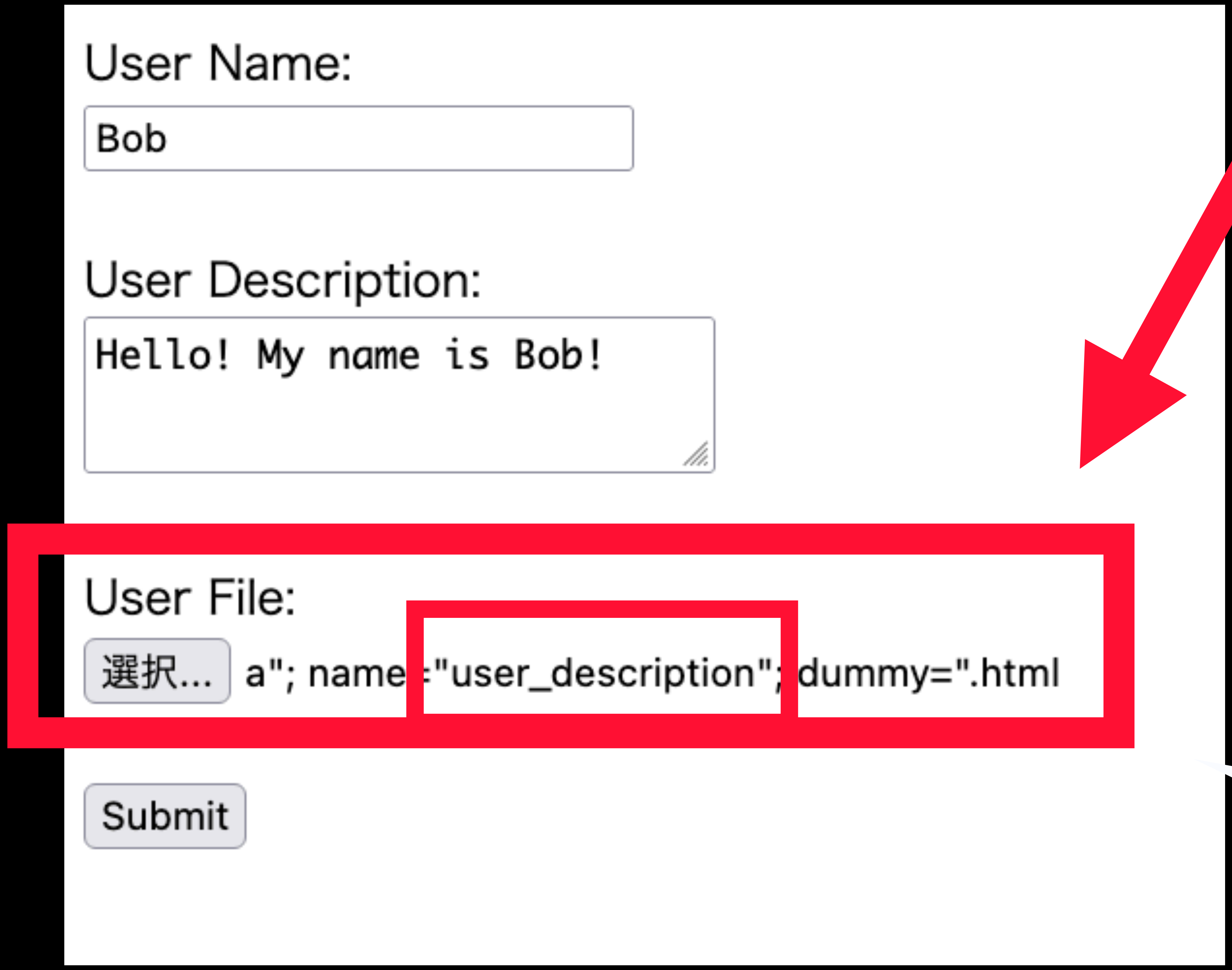
user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

(File Content)

My name is Mallory.

Attack Scenarios (HTTP Request)

Victim



User Name:

Bob

User Description:

Hello! My name is Bob!

User File:

a"; name="user_description"; dummy=".html

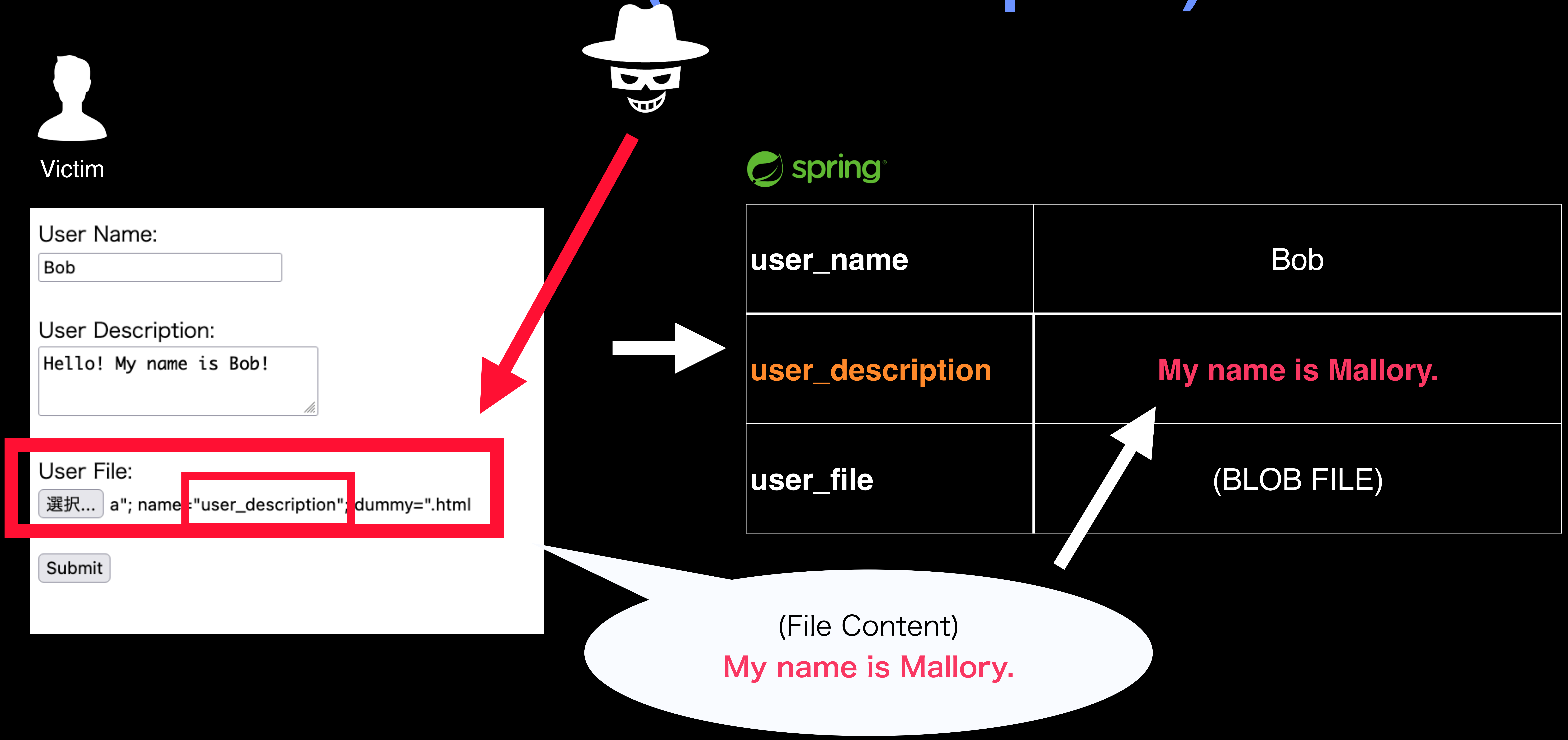
spring

user_name	Bob
user_description	Hello! My name is Bob!
user_file	(BLOB FILE)

(File Content)

My name is Mallory.

Attack Scenarios (HTTP Request)



User Name:
Bob

User Description:
Hello! My name is Bob!

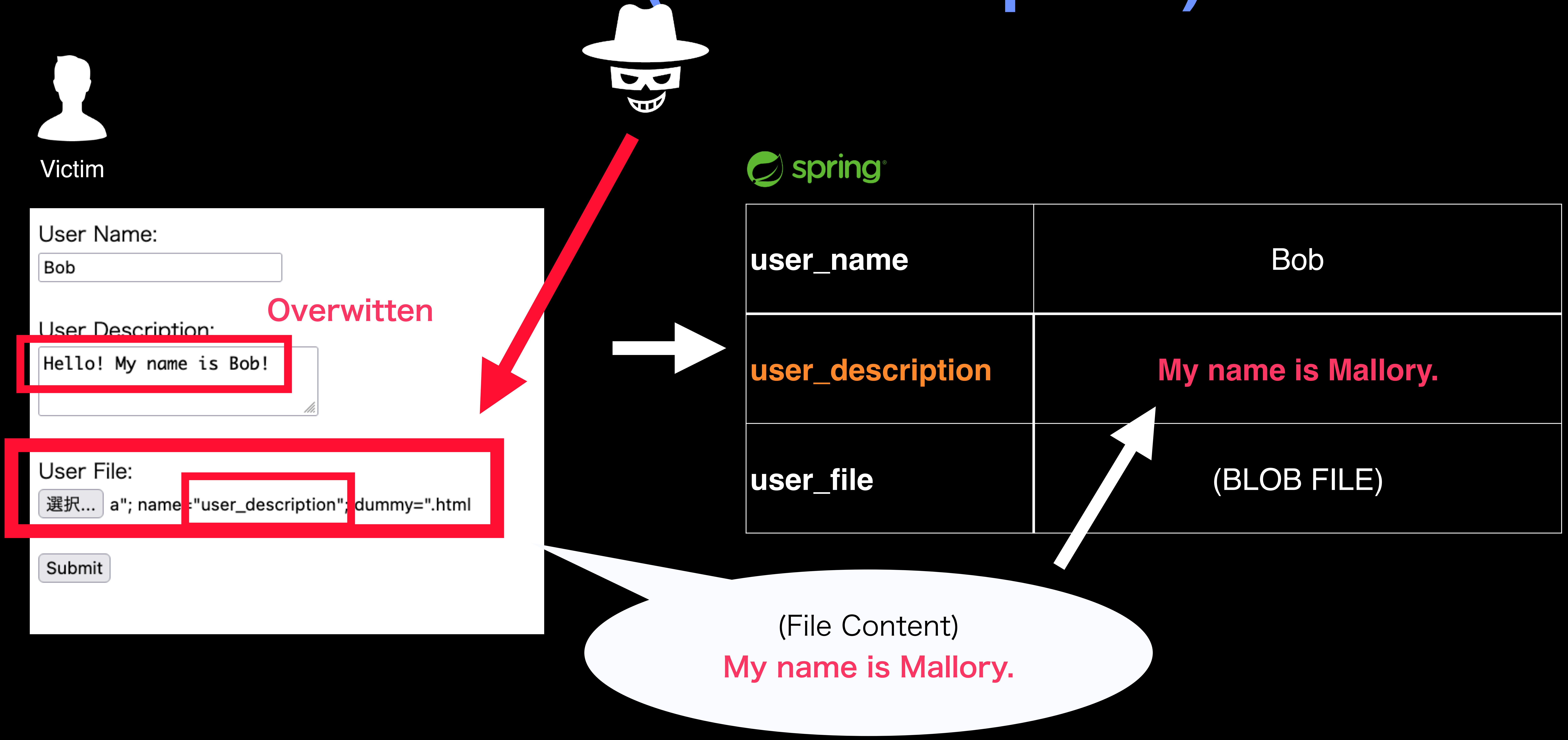
User File:
選択... a"; name="user_description"; dummy=".html

Submit

user_name	Bob
user_description	My name is Mallory.
user_file	(BLOB FILE)

(File Content)
My name is Mallory.

Attack Scenarios (HTTP Request)



Victim

User Name:

Bob

User Description:

Hello! My name is Bob!

Overwritten

User File:

選択... a"; name="user_description"; dummy=".html

Submit

spring

user_name	Bob
user_description	**My name is Mallory.**
user_file	(BLOB FILE)

(File Content)

My name is Mallory.

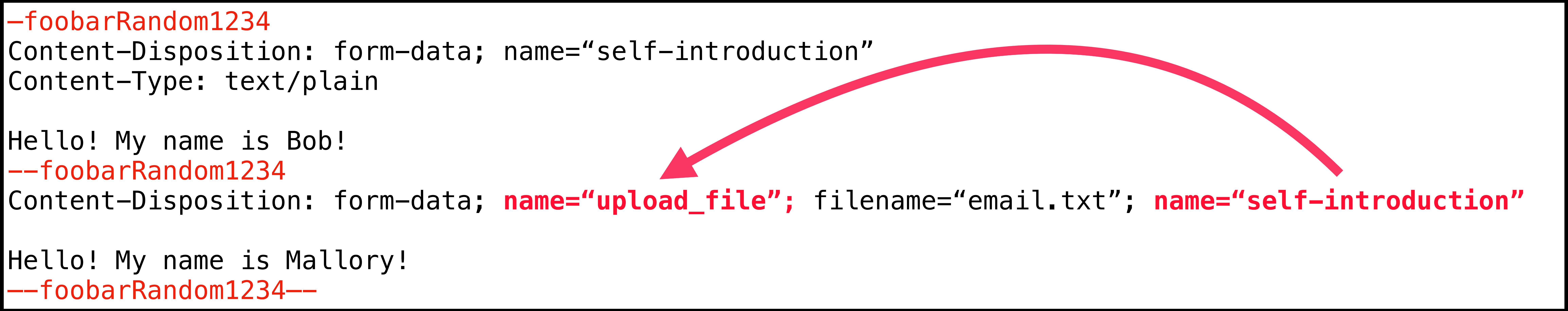
multipart & Content-Disposition

- **Pattern of duplicate fields in one Part (overwriting own fields)**
- **Pattern of duplicated names in multiple Parts (overwriting another field)**

```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction”
Content-Type: text/plain

Hello! My name is Bob!
--foobarRandom1234
Content-Disposition: form-data; name=“upload_file”; filename=“email.txt”; name=“self-introduction”

Hello! My name is Mallory!
--foobarRandom1234--
```

multipart & Content-Disposition

- Pattern of duplicate fields in one Part (overwriting own fields)
- Pattern of duplicated names in multiple Parts
  (overwriting another field)



```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction”
Content-Type: text/plain

Hello! My name is Bob!
--foobarRandom1234
Content-Disposition: form-data; name=“upload_file”; filename=“email.txt”; name=“self-introduction”

Hello! My name is Mallory!
--foobarRandom1234--
```

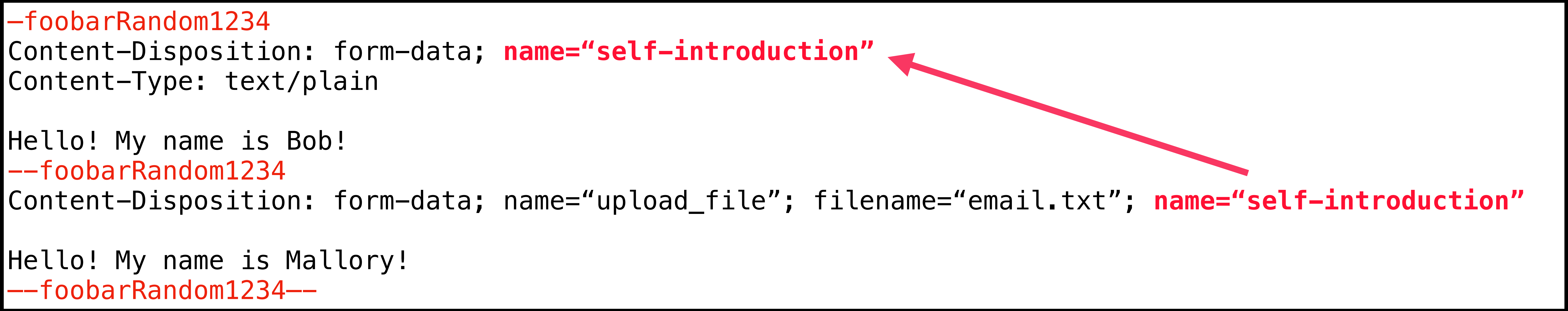
multipart & Content-Disposition

- **Pattern of duplicate fields in one Part (overwriting own fields)**
- **Pattern of duplicated names in multiple Parts (overwriting another field)**

```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction”
Content-Type: text/plain

Hello! My name is Bob!
--foobarRandom1234
Content-Disposition: form-data; name=“upload_file”; filename=“email.txt”; name=“self-introduction”

Hello! My name is Mallory!
--foobarRandom1234--
```

multipart & Content-Disposition

- Pattern of duplicate fields in one Part (overwriting own fields)
- Pattern of duplicated names in multiple Parts (overwriting another field)



```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction”
Content-Type: text/plain

Hello! My name is Bob!
--foobarRandom1234
Content-Disposition: form-data; name=“upload_file”; filename=“email.txt”; name=“self-introduction”

Hello! My name is Mallory!
--foobarRandom1234--
```

multipart & Content-Disposition

- Pattern of duplicate fields in one Part (overwriting own fields)
- Pattern of duplicated names in multiple Parts (overwriting another field)

```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction”
Content-Type: text/plain

Hello! My name is Bob!
--foobarRandom1234
Content-Disposition: form-d                              email.txt”; name=“self-introduction”

Hello! My name is Mallory!
--foobarRandom1234--
```

Self-introduction:
(overwrite)
Hello! My name is Mallory!

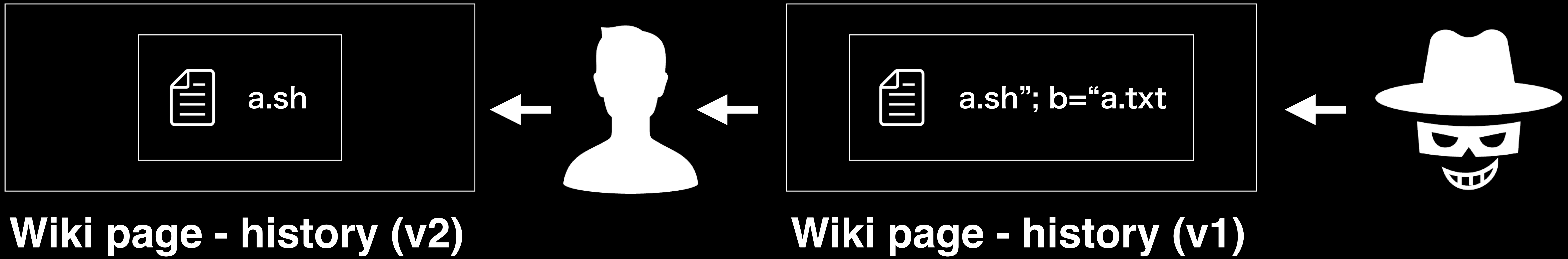
Attack Conditions

- **The condition of the attack** is that the attacker must be able to control the filename of the victim's request.
  - **Uploaded files, URL parameters, and Stored filenames are reflected in the victim's form.**

Attack Conditions

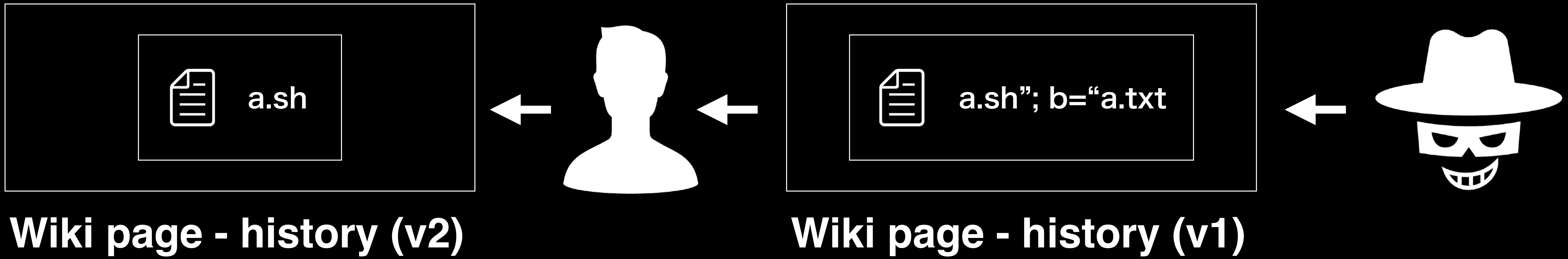
- **The condition of the attack** is that the attacker must be able to **control the filename** of the victim's request.

  - **Uploaded files, URL parameters, and Stored filenames are reflected in the victim's form.**



Attack Conditions

- **The condition of the attack** is that the attacker must be able to **control the filename** of the victim's request.
    - **Uploaded files, URL parameters, and Stored filenames are reflected in the victim's form.**



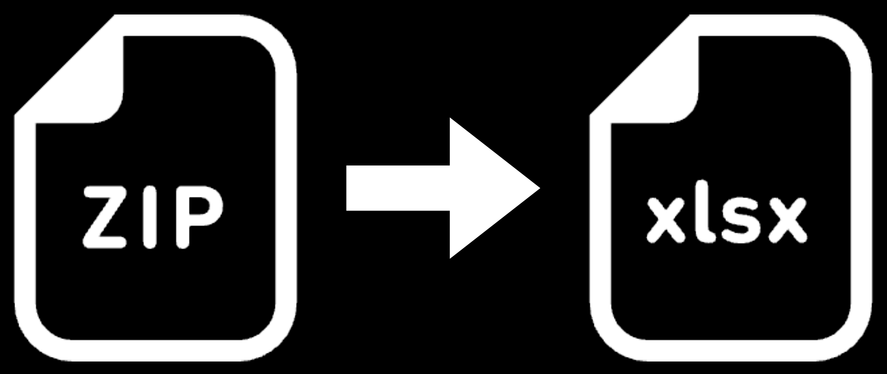
Perhaps there aren't that many such forms in the world.

Affect

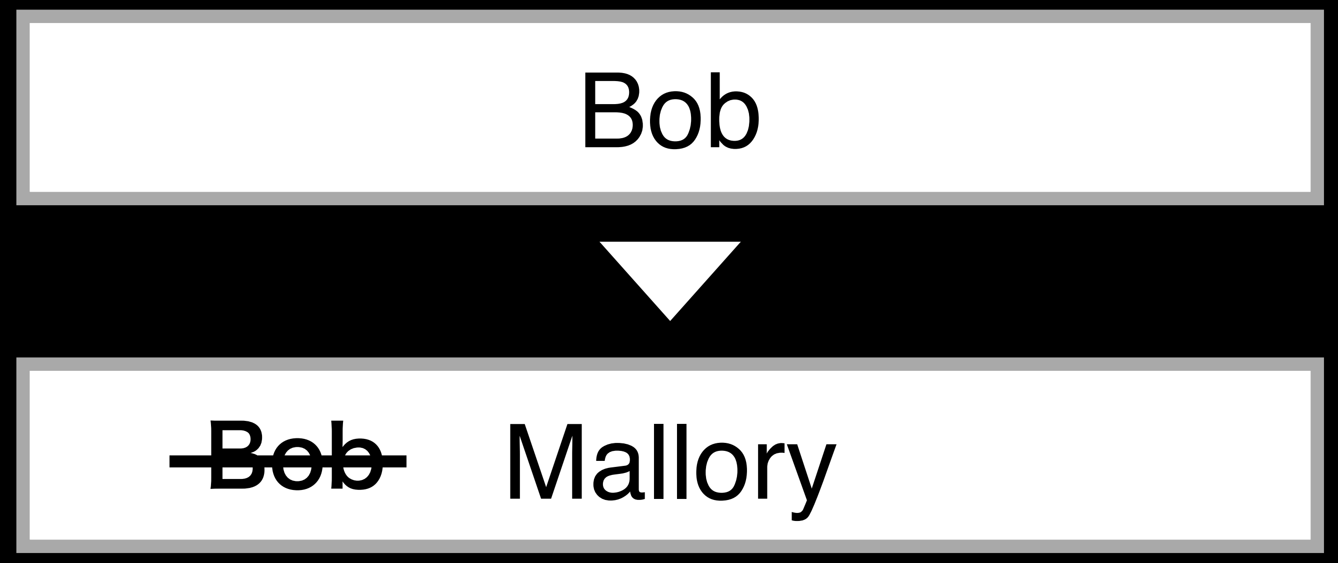
1. **Change the file name / extension.**

2. **Overwriting the values of other fields (part)**
   (Change duplicate “name” field)

3. **Potential DoS problems.**
   (Change “size” field, etc to incorrect value.)

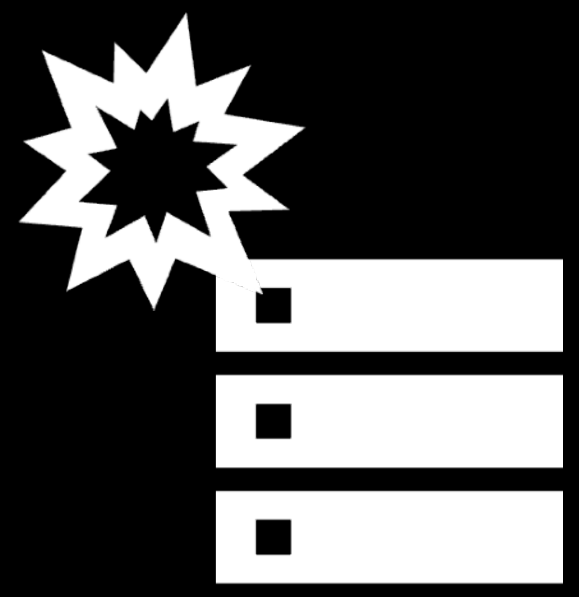


①


Bob

~~Bob~~ Mallory

②



③

(False) Report to Spring

Sending this file allows Spring to tamper with the values.

a.sh"; name=“dummy"; filename="dummy"; size=1234; dummy=".txt

(False) Report to Spring

Sending this file allows Spring to tamper with the values.

📄 **a.sh"; name=“dummy"; filename="dummy"; size=1234; dummy=".txt**

Isn't that the problem with the Content-Disposition generation part of the Firefox?

spring

(False) Report to Spring

Sending this file allows Spring to tamper with the values.

```
a.sh"; name=“dummy"; filename="dummy"; size=1234; dummy=".txt
```

Isn't that the problem with the Content-Disposition generation part of the Firefox?

spring

🙇

Report to Firefox

Firefox

This is a Firefox security issue.

https://bugzilla.mozilla.org/show_bug.cgi?id=1556711

Report to Firefox

Firefox

This is a Firefox security issue.

https://bugzilla.mozilla.org/show_bug.cgi?id=1556711

🕺

Standards and Guideline

- The RFC and WHATWG's html spec
  do not list any escaping **requirements** for this issue.

  - **RFC**
    - **As of 2023, there is still no clear statement of escape requirement**

  - **WHATWG html spec > multipart/form-data**
    - Prior to 2021, it was stated that "Double-Quote, NewLine may be deleted."
    - <u>The content added in 2021 included an escape requirement as a mandatory item.</u>
      - “ —> %22
      - \r —> %0D
      - \n —> %0A

  - **OWASP**
    - **Under discussion in OWASP ASVS v5 as of 2023**

It’s a happy ending.

It was not
~~It's a~~ happy ending.

A lot happened

Firefox

- **It wasn't fix for 2.5 years. (2019 —> 2021)**
- **The fix was also a silent fix (2021).**
- (No CVE was also issued. 😭)

A lot happened

Firefox

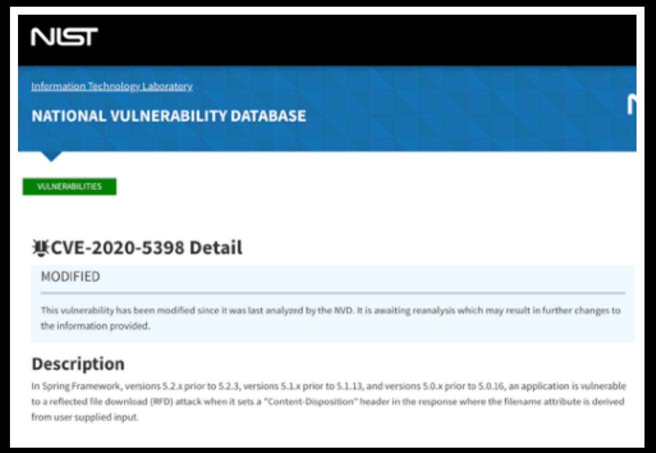
- **It wasn't fix for 2.5 years. (2019 —> 2021)**
- **The fix was also a silent fix (2021).**
- (No CVE was also issued. 😭)

Spring

- **CVE issued around Content-Disposition, which I thought was "not vulnerable”. (Lack of Tenacity)**

Spring RFD Vuln

https://nvd.nist.gov/vuln/detail/CVE-2020-5398



NIST
Information Technology Laboratory
NATIONAL VULNERABILITY DATABASE
VULNERABILITIES

CVE-2020-5398 Detail

MODIFIED

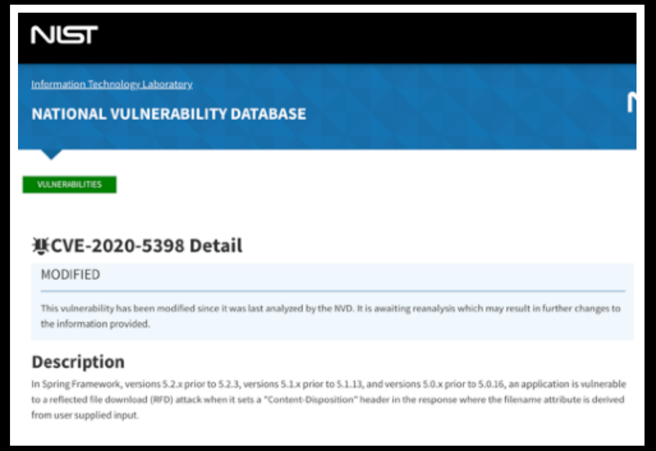
This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Description

In Spring Framework, versions 5.2.x prior to 5.2.3, versions 5.1.x prior to 5.1.13, and versions 5.0.x prior to 5.0.16, an application is vulnerable to a reflected file download (RFD) attack when it sets a "Content-Disposition" header in the response where the filename attribute is derived from user supplied input.

Spring RFD Vuln

https://nvd.nist.gov/vuln/detail/CVE-2020-5398


NIST
Information Technology Laboratory
NATIONAL VULNERABILITY DATABASE
VULNERABILITIES

CVE-2020-5398 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

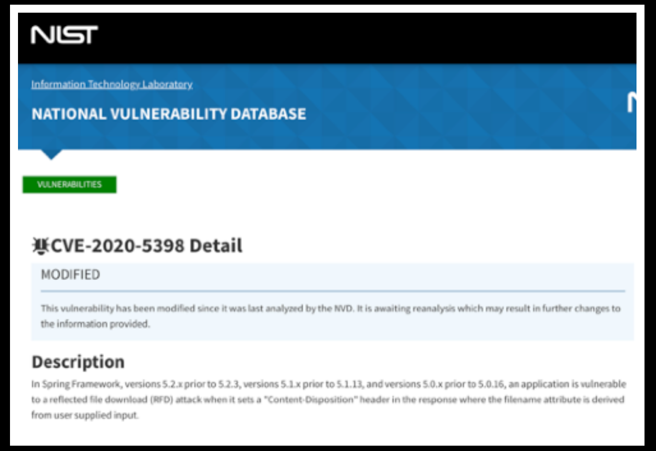
Description

In Spring Framework, versions 5.2.x prior to 5.2.3, versions 5.1.x prior to 5.1.13, and versions 5.0.x prior to 5.0.16, an application is vulnerable to a reflected file download (RFD) attack when it sets a "Content-Disposition" header in the response where the filename attribute is derived from user supplied input.

In Spring Framework, versions 5.2.x prior to 5.2.3 ……, an application is vulnerable to a reflected file download (RFD) attack when it sets a "Content-Disposition" header in the response where the filename attribute is derived from user supplied input.

Spring RFD Vuln

https://nvd.nist.gov/vuln/detail/CVE-2020-5398



NIST
Information Technology Laboratory
NATIONAL VULNERABILITY DATABASE
VULNERABILITIES

CVE-2020-5398 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Description

In Spring Framework, versions 5.2.x prior to 5.2.3, versions 5.1.x prior to 5.1.13, and versions 5.0.x prior to 5.0.16, an application is vulnerable to a reflected file download (RFD) attack when it sets a "Content-Disposition" header in the response where the filename attribute is derived from user supplied input.

In Spring Framework, versions 5.2.x prior to 5.2.3 ……, an application is vulnerable to a **reflected file download (RFD) attack when it sets a "Content-Disposition" header in the response where the filename attribute is derived from user supplied input.**

I wasn’t aware of this when I was doing my research.

What is RFD ?

RFD is a vulnerability that creates a URL from a trusted domain
(e.g., google.com) that allows the victim to download a file that the attacker can control.

The victim assumes that the file is harmless because it is on google.com, and downloads and executes the attacker-controlled malware.

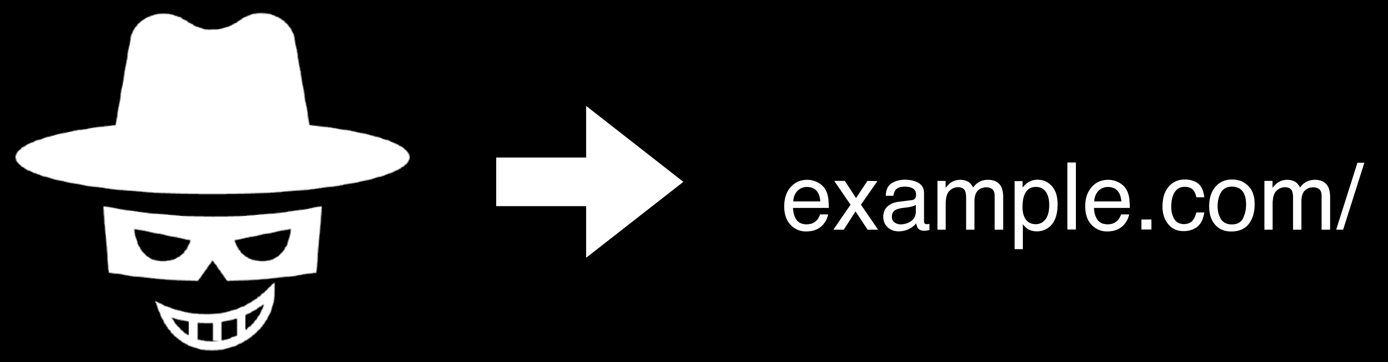
https://www.blackhat.com/docs/eu-14/materials/eu-14-Hafif-Reflected-File-Download-A-New-Web-Attack-Vector.pdf

What is RFD ?

RFD is a vulnerability that creates a URL from a trusted domain
(e.g., google.com) that allows the victim to download a file that the attacker can control.

The victim assumes that the file is harmless because it is on google.com, and downloads and executes the attacker-controlled malware.

https://www.blackhat.com/docs/eu-14/materials/eu-14-Hafif-Reflected-File-Download-A-New-Web-Attack-Vector.pdf



example.com/

What is RFD ?

RFD is a vulnerability that creates a URL from a trusted domain
(e.g., google.com) that allows the victim to download a file that the attacker can control.

The victim assumes that the file is harmless because it is on google.com, and downloads and executes the attacker-controlled malware.

https://www.blackhat.com/docs/eu-14/materials/eu-14-Hafif-Reflected-File-Download-A-New-Web-Attack-Vector.pdf

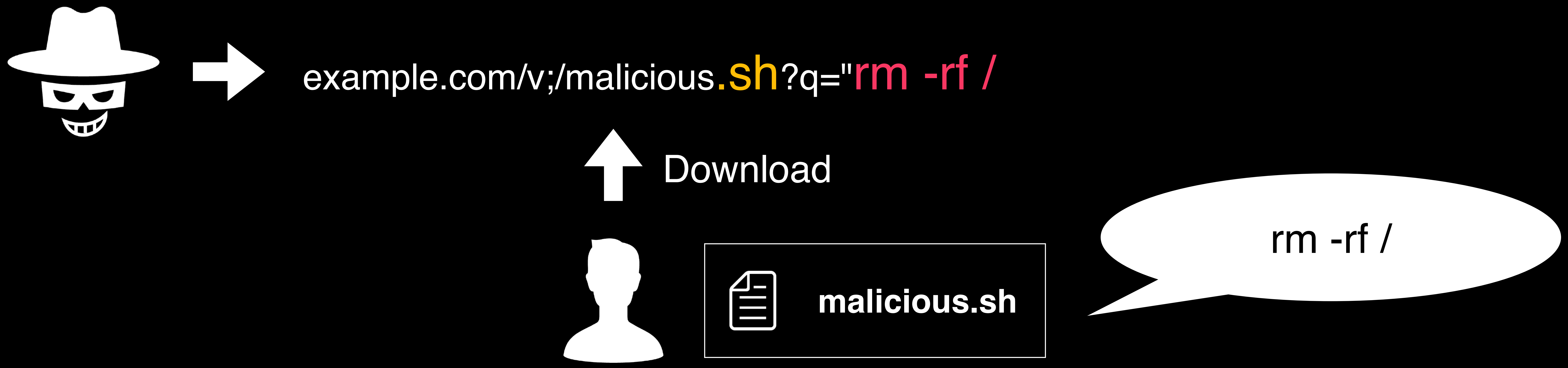
→ example.com/v;/malicious.sh?q="rm -rf /

What is RFD ?

RFD is a vulnerability that creates a URL from a trusted domain
(e.g., google.com) that allows the victim to download a file that the attacker can control.

The victim assumes that the file is harmless because it is on google.com, and downloads and executes the attacker-controlled malware.

https://www.blackhat.com/docs/eu-14/materials/eu-14-Hafif-Reflected-File-Download-A-New-Web-Attack-Vector.pdf



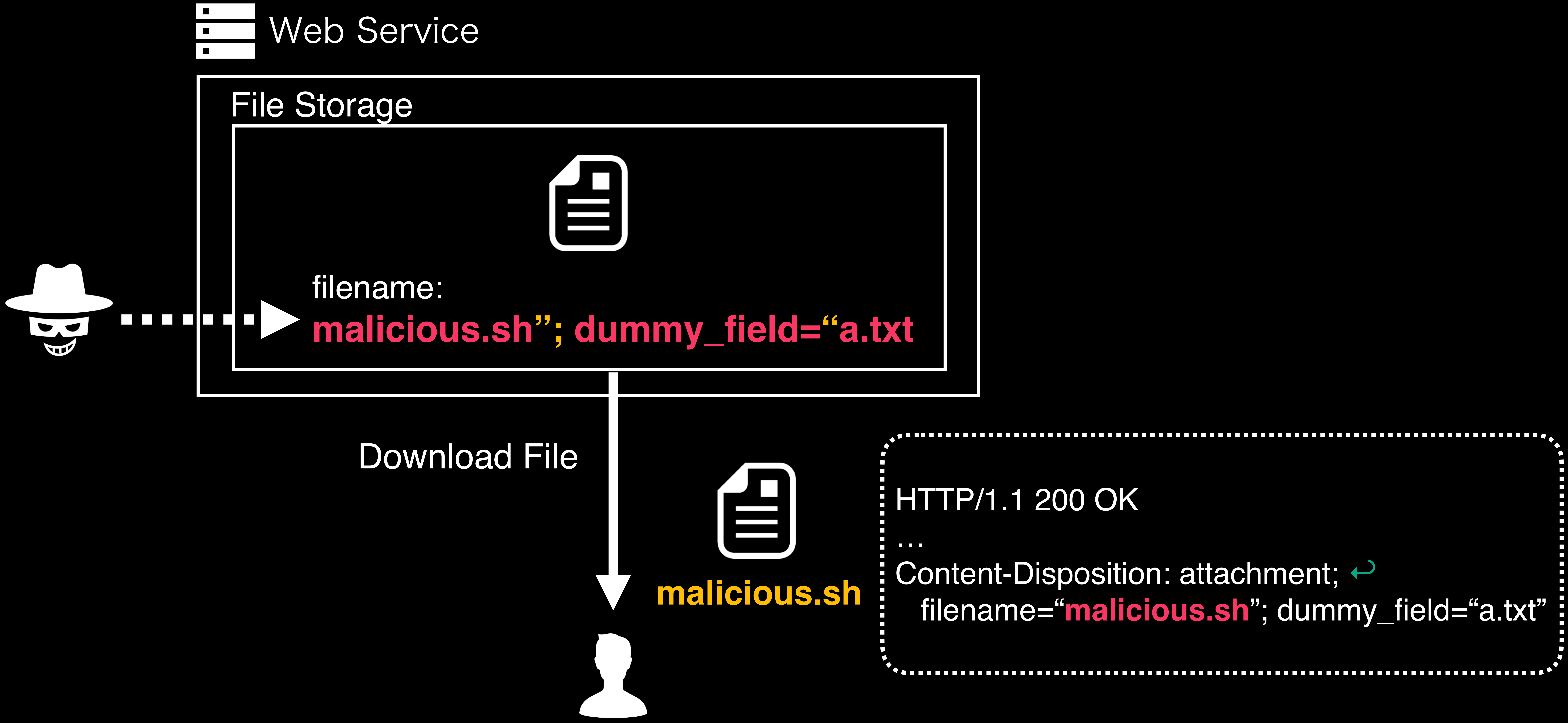
example.com/v;/malicious.sh?q="rm -rf /

Download

malicious.sh

rm -rf /

What is RFD ?



Web Service

File Storage

filename:
malicious.sh”; dummy_field=“a.txt

Download File

malicious.sh

```
HTTP/1.1 200 OK
…
Content-Disposition: attachment; ↩
  filename=“malicious.sh”; dummy_field=“a.txt”
```

Trying Again

This time I will do my best to find out.

Overall evaluation of the area where Content-Disposition appears.

- Research Scope [Request, Response, Email]
- Examine the root cause of this problem.
- Assessing the impact of a problem depending on where it occurs

System type (Research Scope)

Request
(Multipart)

Email
(Multipart)

Response

System type (Research Scope)

Request
(Multipart)

Browser

HTTP Client

Email
(Multipart)

Mail Library
(Mail Client)

Response

Web Framework

Organizing the Situation (Root Cause)

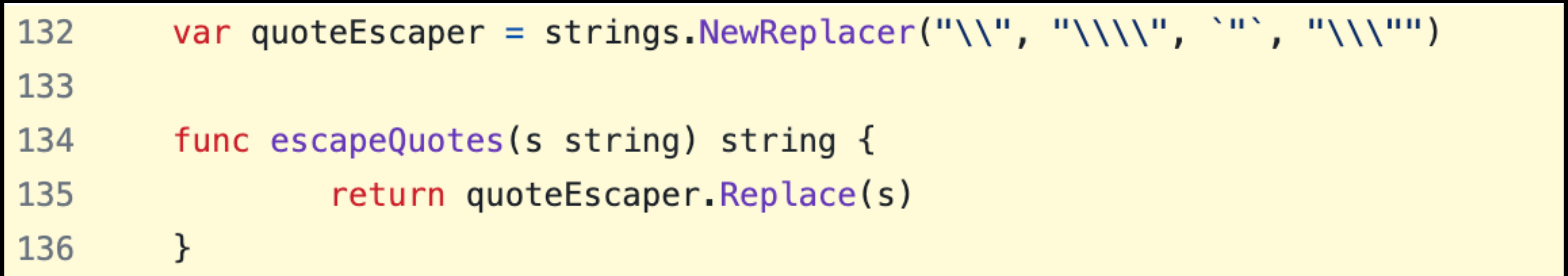
The vulnerability occurs when "filename" is not escaped.

- “ —> %22
- \r —> %0d
- \n —> %0a

WHATWG’s HTML Spec > multipart/form-data

- “ —> \”
- \ —> \\

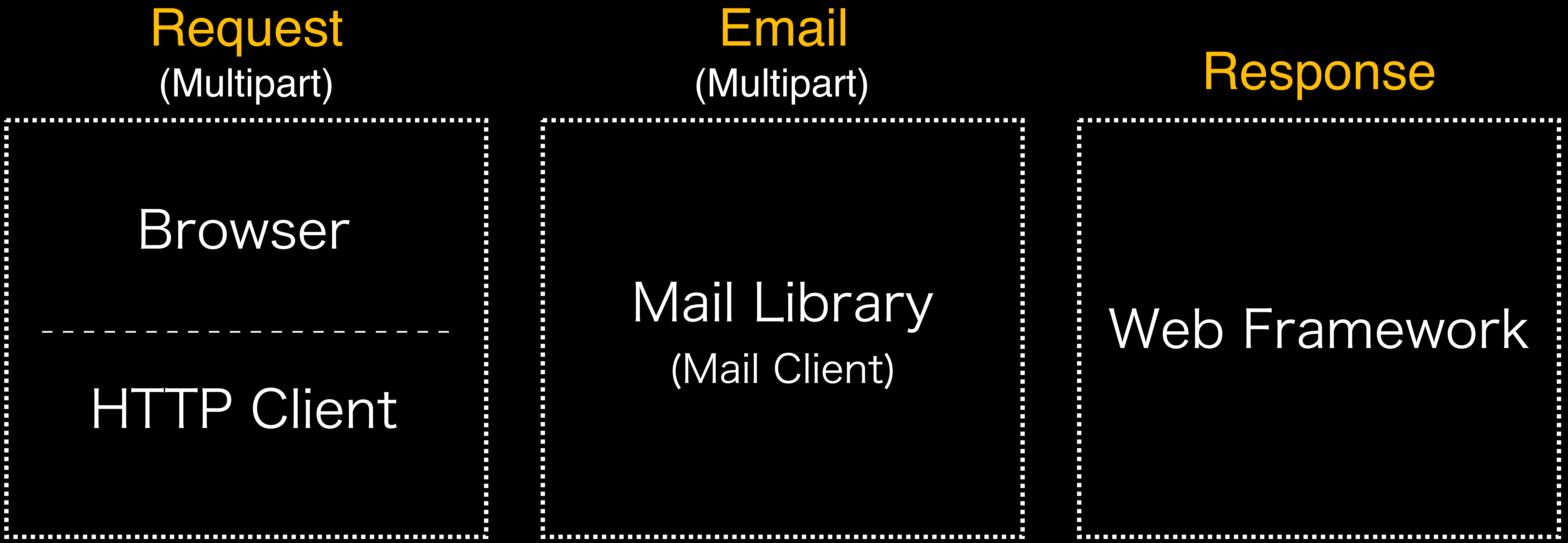
Countermeasures on Golang & some application
(Unique Methods)



```go
var quoteEscaper = strings.NewReplacer("\\", "\\\\", `"`, "\\\"")

func escapeQuotes(s string) string {
	return quoteEscaper.Replace(s)
}
```

https://github.com/golang/go/blob/561a5079057e3a660ab638e1ba957a96c4ff3fd1/src/mime/multipart/writer.go#L132-L136

System type (Research Scope)



Request
(Multipart)

Browser

HTTP Client

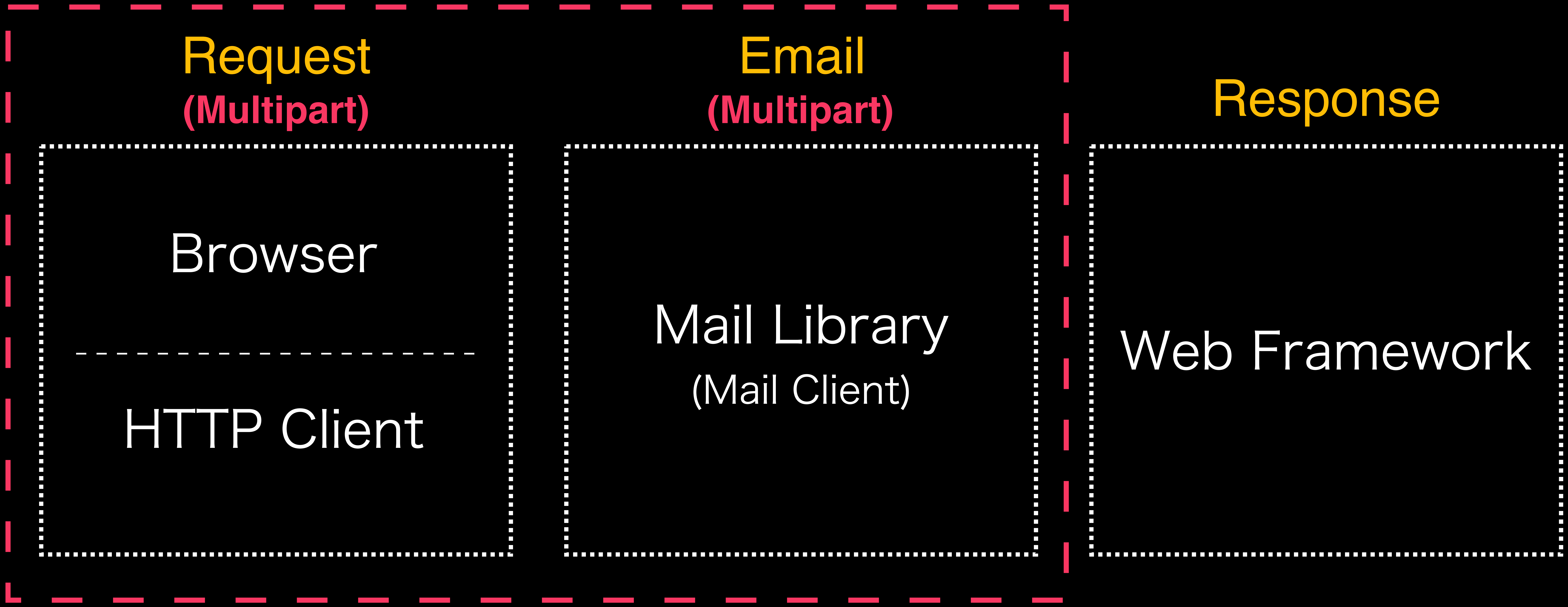
Email
(Multipart)

Mail Library
(Mail Client)

Response

Web Framework

System type (Research Scope)



Request

(Multipart)

Browser

HTTP Client

Email

(Multipart)

Mail Library

(Mail Client)

Response

Web Framework

Multipart (HTTP Req / Email)

```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“hello.txt”
Content-Type: text/plain

Hello! My name is Bob!

––foobarRandom1234––
```

Multipart (HTTP Req / Email)

Hello! My name is Bob!

a.sh“; b=“ hello.txt

```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“a.sh”; b=”hello.txt”
Content-Type: text/plain

Hello! My name is Bob!

--foobarRandom1234--
```

Multipart (HTTP Req / Email)

hello.txt

```
–foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“hello.txt”
Content-Type: text/plain

Hello! My name is Bob!

––foobarRandom1234––
```

Multipart (HTTP Req / Email)

“\r\n\r\n hello.txt

```
—foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“”

hello.txt”
Content-Type: text/plain

Hello! My name is Bob!

--foobarRandom1234--
```

Multipart (HTTP Req / Email)

Hello! My name is Bob!

“\r\n\r\n hello.txt

```
-foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“”

hello.txt”
Content-Type: text/plain

Hello! My name is Bob!

--foobarRandom1234--
```

New Line

New Line

File Body (Inject Content)

Multipart (HTTP Req / Email)

a.sh“;\r\n Content-Type: application/x-sh\r\n\r\n#!/bin/bash\r\nrm -rf /

```
-foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“a.sh”
Content-Type: application/x-sh

#!/bin/bash
rm -rf /

Content-Type: text/plain

Hello! My name is Bob!

--foobarRandom1234--
```

Multipart (HTTP Req / Email)

Hello! My name is Bob!

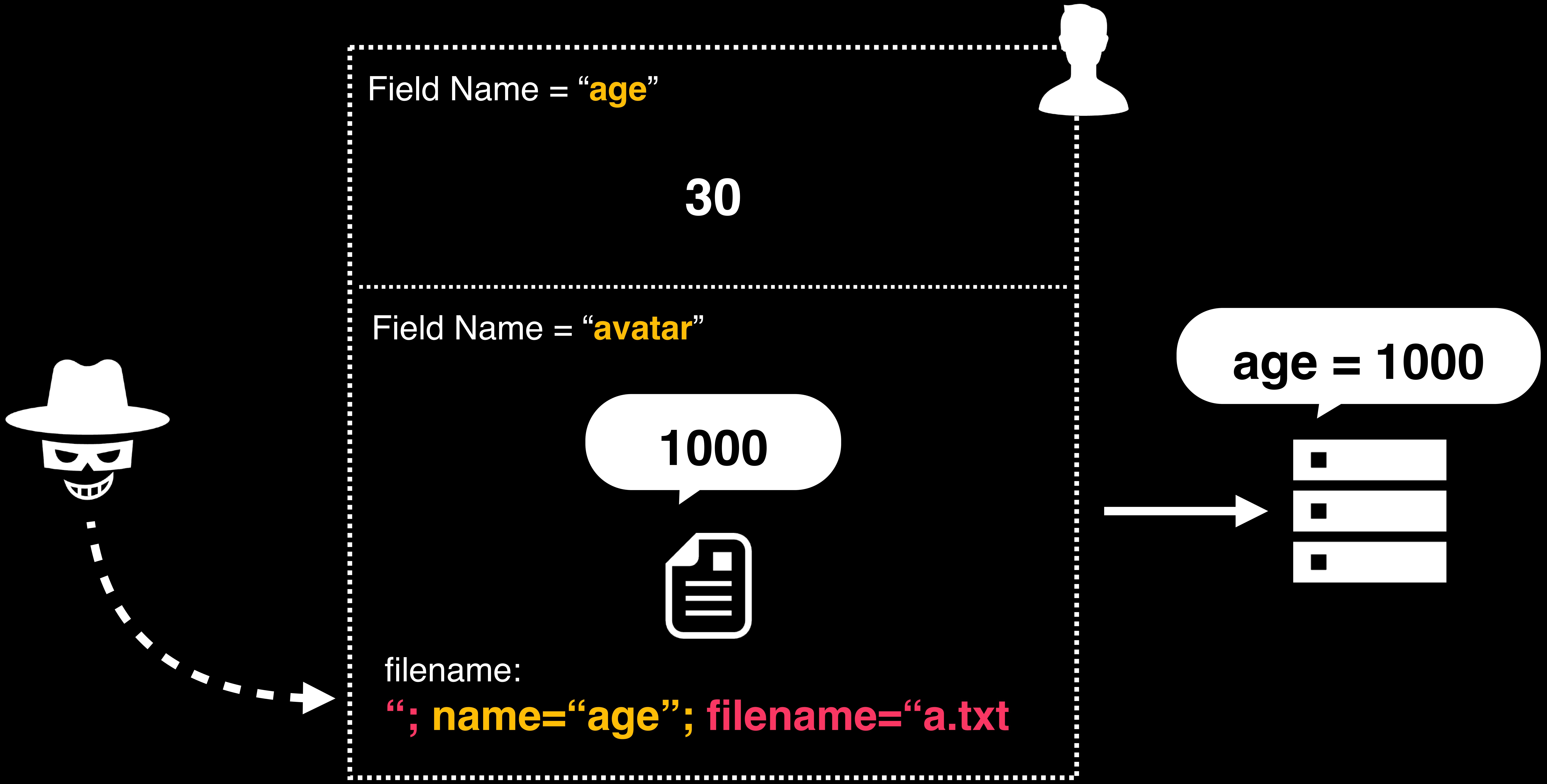
a.sh“;\r\n Content-Type: application/x-sh\r\n\r\n#!/bin/bash\r\nrm -rf /

```
-foobarRandom1234
Content-Disposition: form-data; name=“self-introduction” filename=“a.sh”
Content-Type: application/x-sh

#!/bin/bash
rm -rf /

Content-Type: text/plain

Hello! My name is Bob!

--foobarRandom1234--
```

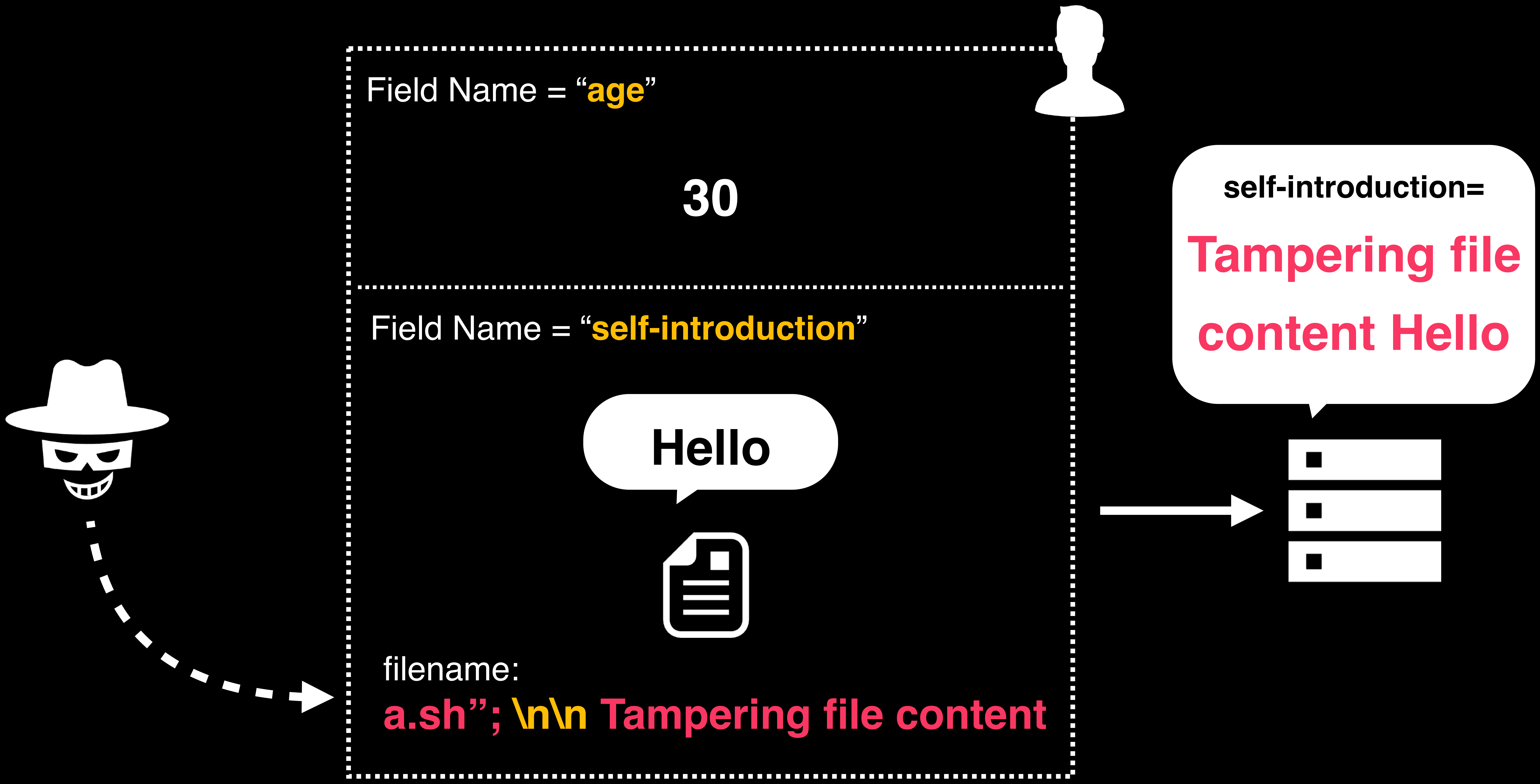
Tamper Content-Type.

Tamper txt to shell.

Scenario - Tampering Request



Scenario - Tampering Request



Field Name = “**age**”

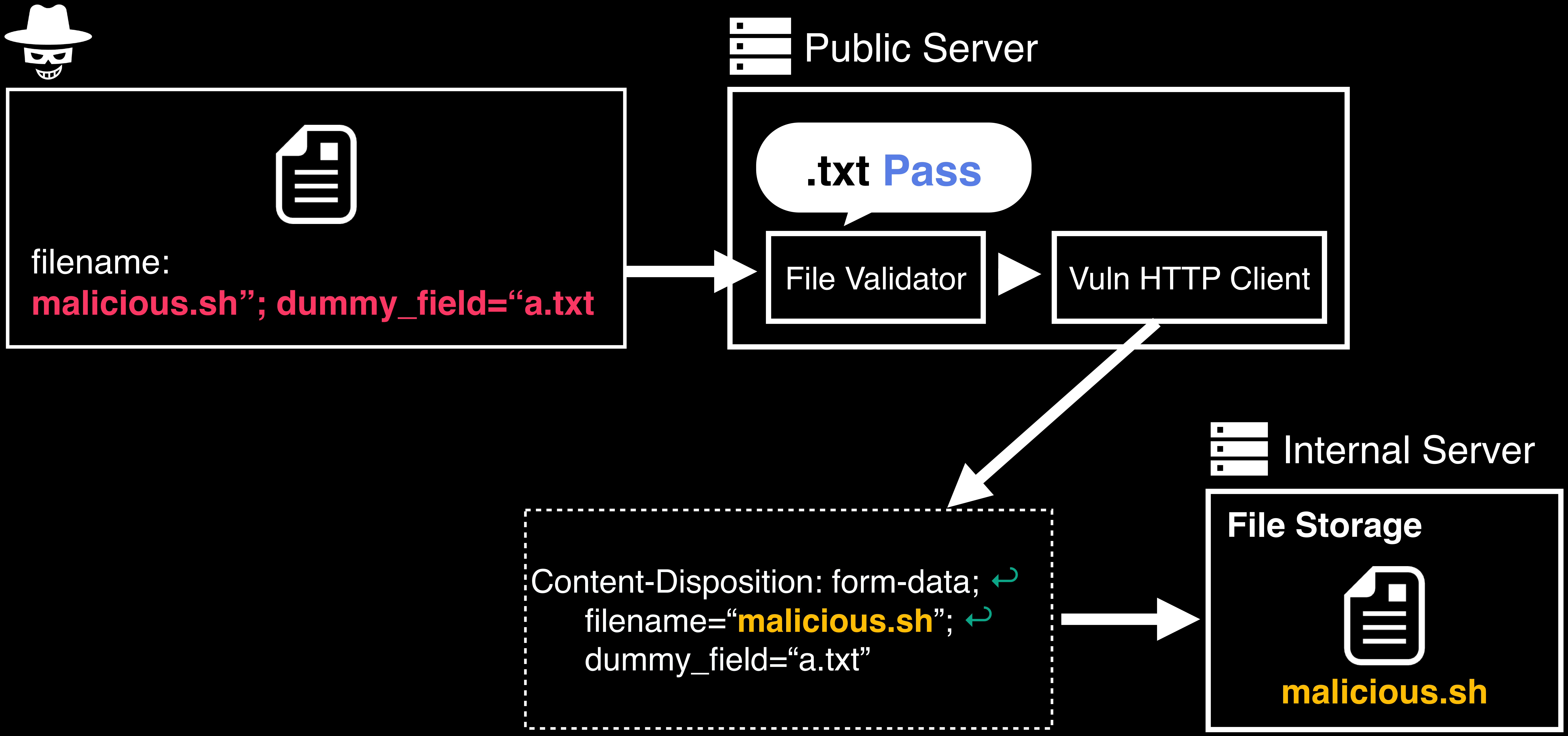
30

Field Name = “**self-introduction**”

filename:

a.sh”; \n\n Tampering file content

Scenario - Validation Bypass - HTTP Client



filename:
malicious.sh”; dummy_field=“a.txt

Public Server

.txt Pass

File Validator

Vuln HTTP Client

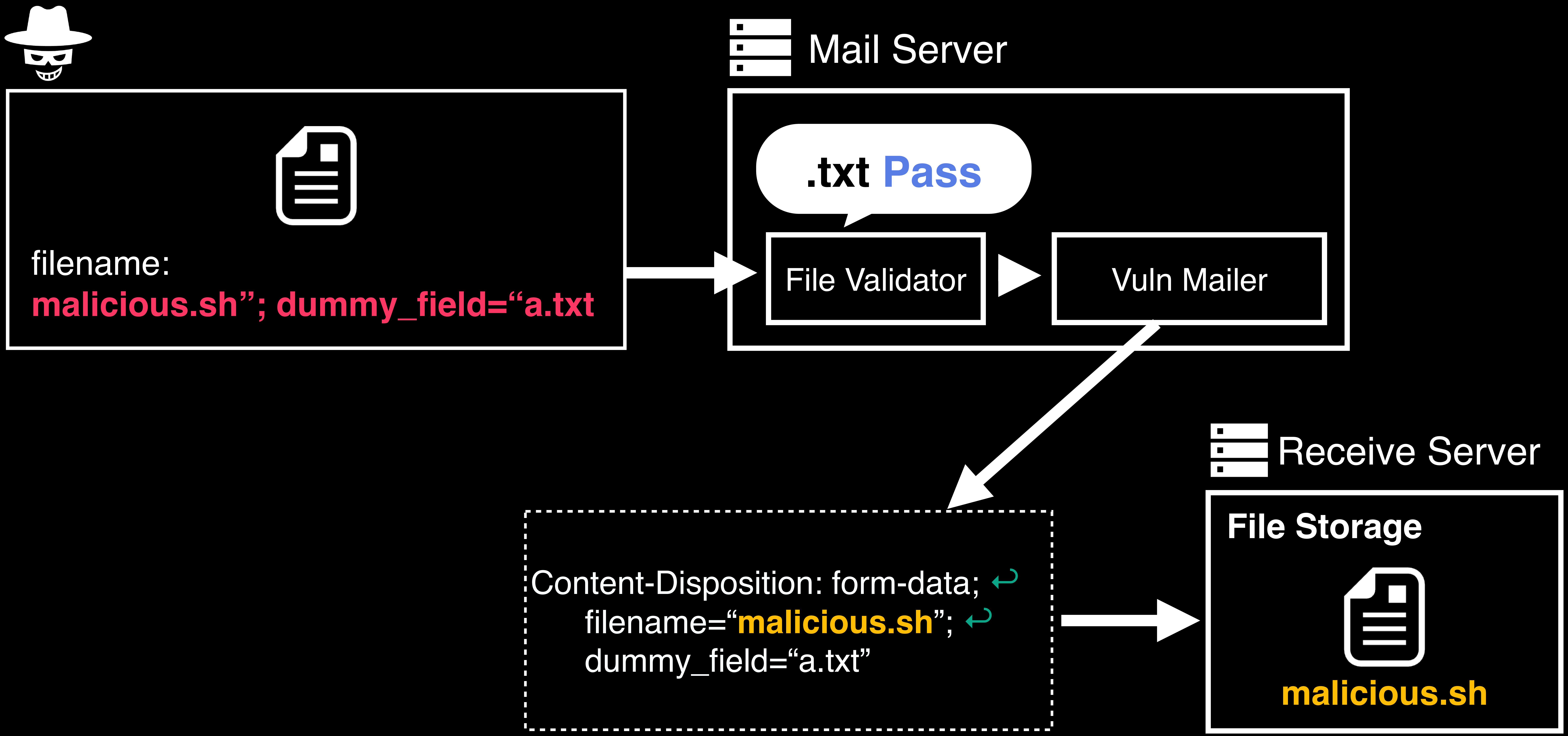
```
Content-Disposition: form-data; ↩
      filename=“malicious.sh”; ↩
      dummy_field=“a.txt”
```

Internal Server

File Storage

malicious.sh

Scenario - Validation Bypass - Mailer



filename:
malicious.sh"; dummy_field="a.txt

Mail Server

.txt Pass

File Validator

Vuln Mailer

```
Content-Disposition: form-data; ↩
      filename="malicious.sh"; ↩
      dummy_field="a.txt"
```

Receive Server

File Storage

malicious.sh

Attack Scenario - Multipart

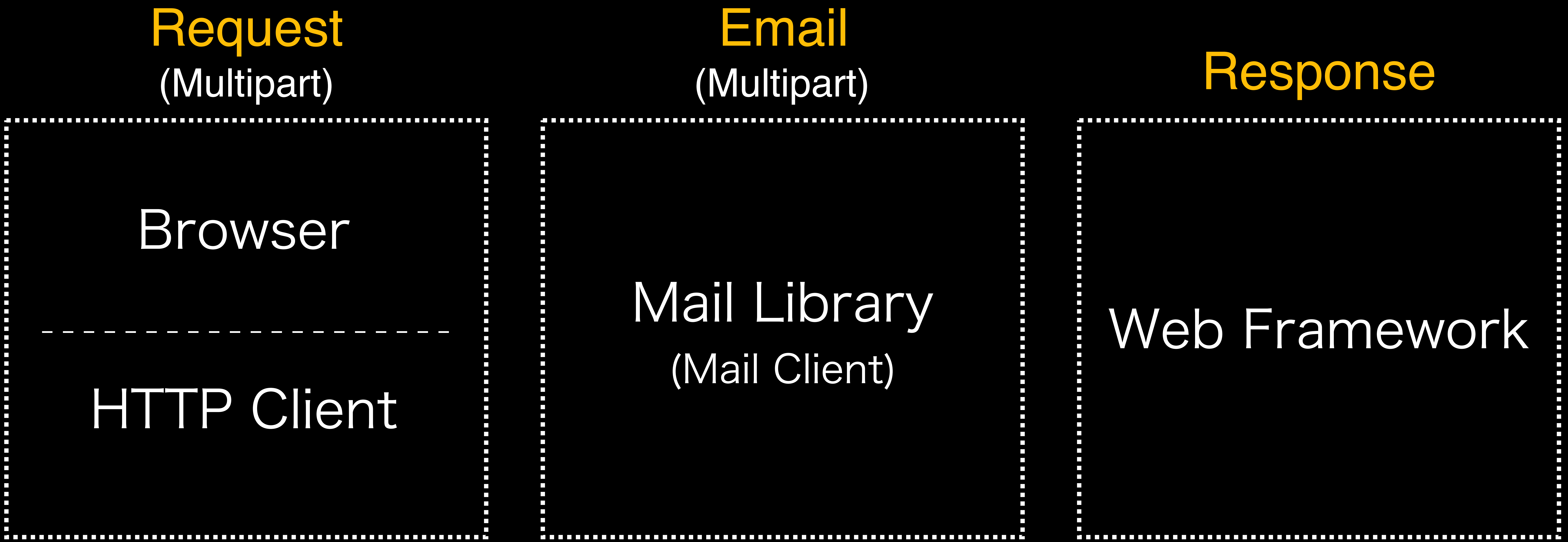
- Request Tampering

  - Overriding other fields
  - Append to file contents
  - Potential DoS - Injecting invalid Size / Creation-Date / etc

- Validation Bypass

  - (If multipart generation occurs after file validation)
    It is possible to bypass validation.

System type (Research Scope)



Request
(Multipart)

Browser

HTTP Client

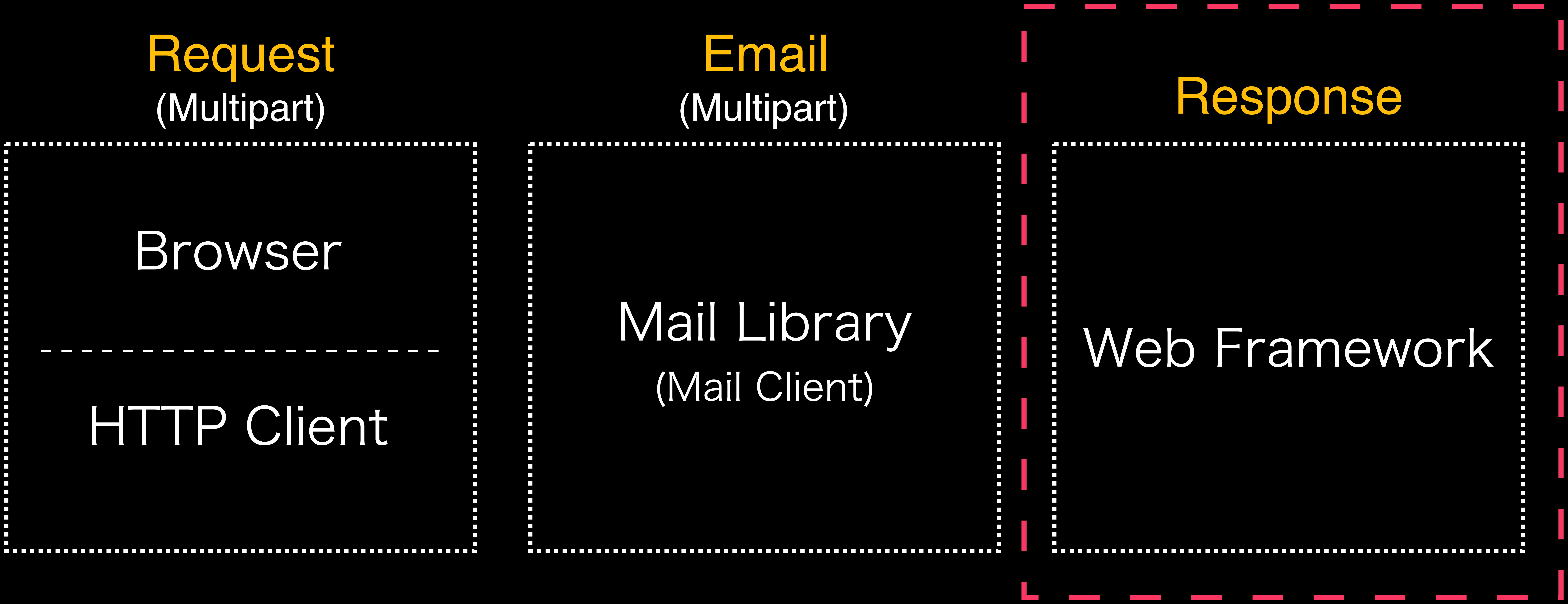
Email
(Multipart)

Mail Library
(Mail Client)

Response

Web Framework

System type (Research Scope)



Request
(Multipart)

Browser

HTTP Client

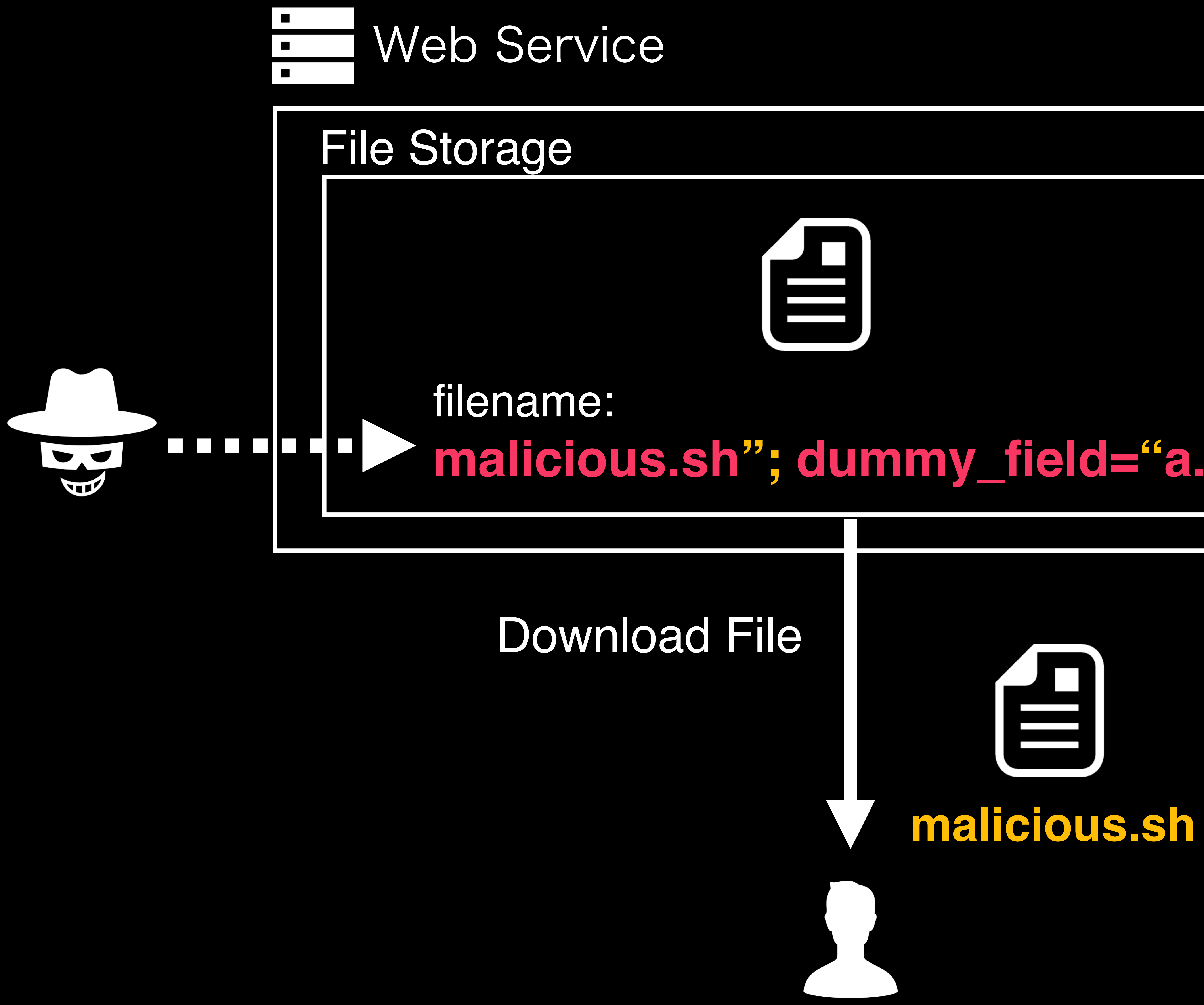
Email
(Multipart)

Mail Library
(Mail Client)

Response

Web Framework

RFD (Reflect File Download) - Response



Web Service

File Storage

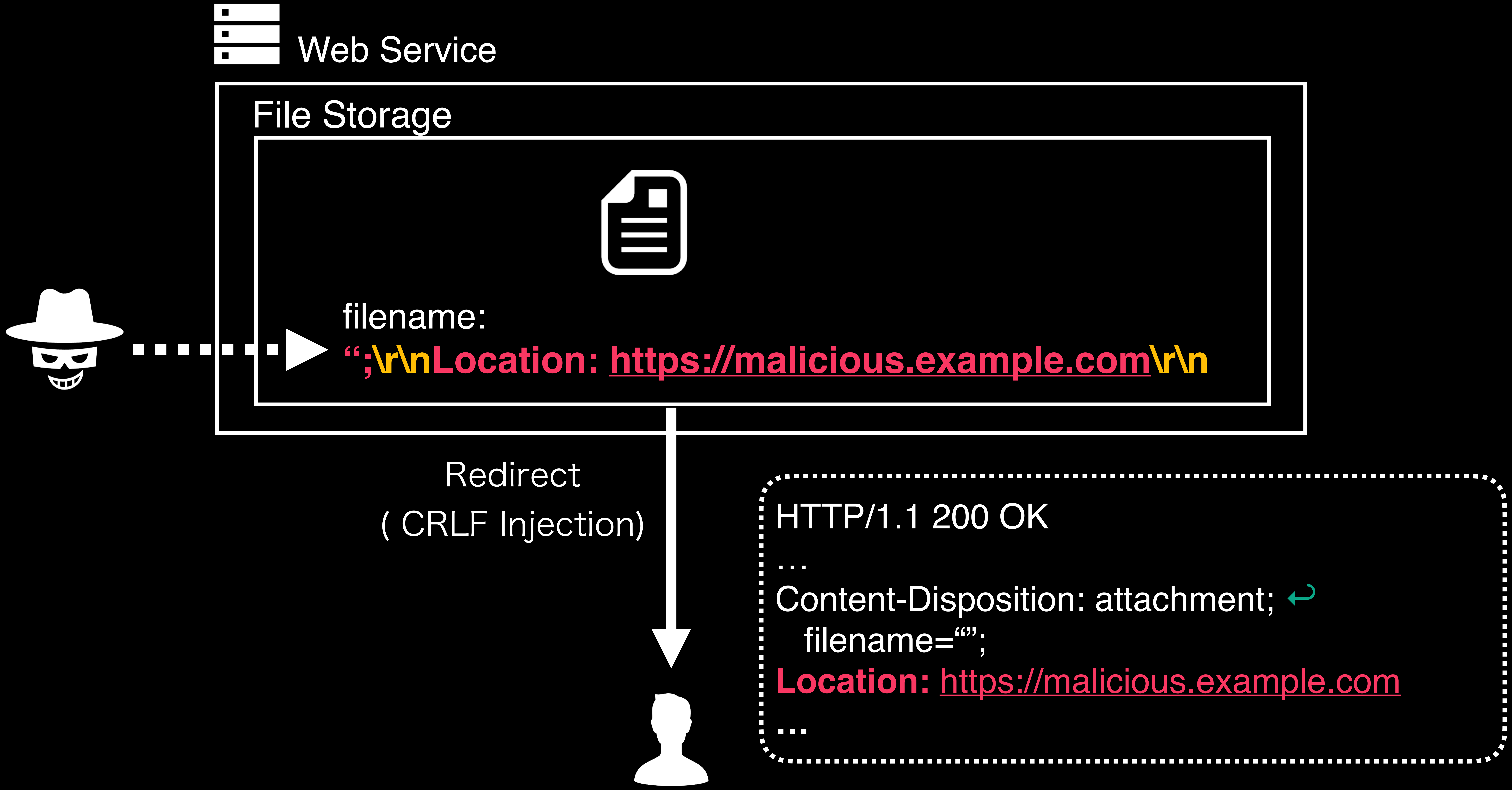
filename:
malicious.sh”; dummy_field=“a.txt

Download File

malicious.sh

```
HTTP/1.1 200 OK
…
Content-Disposition: attachment; ↩
   filename=“malicious.sh”; dummy_field=“a.txt”
```

CRLF Injection - Response



Web Service

File Storage

filename:
“;\r\nLocation: https://malicious.example.com\r\n

Redirect
(CRLF Injection)

```
HTTP/1.1 200 OK
…
Content-Disposition: attachment; ↩
  filename=“”;
Location: https://malicious.example.com
…
```

Research Result

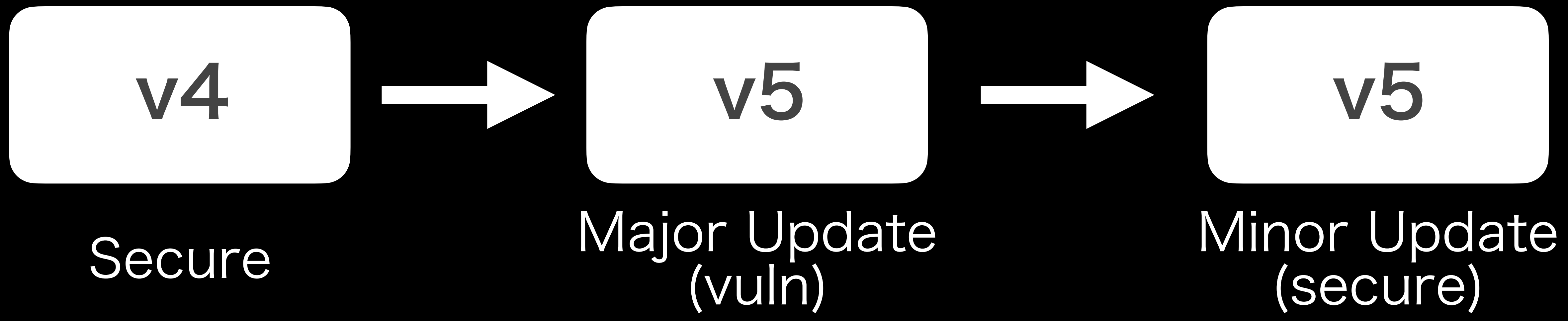
HTTP Request - Research Result

Name	Category	Comment	Github Star	Vuln Num	Lack of escape	Fix
Firefox	Browser			-	-	✅
httpcomponents-client	Java HTTP Client	Caused by regression	1.4k	-	Double-Quote CRLF	✅
httparty	Ruby HTTP Client		5.7k	GHSA-5pq7-52mg-hr42	Double-Quote	✅
Playframework WS	Scala/Java HTTP Client		12.4k	-	Double-Quote CRLF	✅
Fuel	Kotlin HTTP Client	No reply from the maintainer.	1.5k	-	Double-Quote CRLF	
form-data	Node.js MultipartFormData	not maintained & no patch.	2.2k	-	Double-Quote CRLF	
guzzle/psr7	PHP HTTP Client	Maintainer states that	7.7k	-	Double-Quote	

Case: apache/httpcomponents-client

Vulnerability caused by regression

- v4 code was secure (There was no comment at the escape code)
- Major fixes in v5, causing regression
  - Starting with v5, the policy is to not provide a security model for protocols finer than the HTTP Protocol, such as multipart.
  - If the library does not handle protocols like multipart, it may not be called vulnerable.
  - Fixed



Case: Patch Does NOT exist

For some libraries, I’m unable to contact the maintainers and currently no patches exist.

- **kittinunf/Fuel**
  - No response.
  - There was a major v3 update.
  - The Multipart class itself is gone.
- **form-data/form-data**
  - Not maintained.
- **guzzle/psr7**
  - **The maintainer says "User input should not be passed to filename. Therefore, it is not a vulnerability.**

Email - Research Result

Name	Category	Comment	Github Star	Vuln Num	Lack of escape	Fix
Python	Programing Lang		-	-	CRLF	
mikel/mail	Ruby Mail Library		3.5k	-		
go-gomail/ gomail	Go Mail Library	Not maintained	4k	-		

Note:

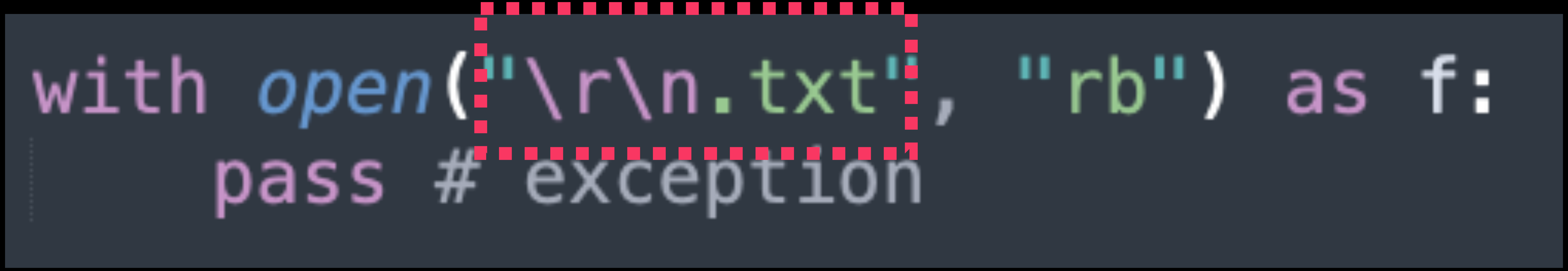
Multipart is also used in the mail storage format called mbox.
However, this is outside the scope of this research.

Case: Python

CRLF (\r\n) is not escaped when creating a mail (multipart).

- **Vulnerabilities Tagged**
- **Impact is quiet limited**
  - When opening a file, files containing (\r\n) are less affected because an Exception is raised.

https://github.com/python/cpython/issues/100612



```
with open("\r\n.txt", "rb") as f:
    pass # exception
```

HTTP Response - Research Result

Name	Category	Comment	Github Star	Vuln Number	Lack of escape	Fix
Sinatra	Ruby Web Framework		12k	CVE-2022-45442	Double-Quote	✅
Iris	Go Web Framework		24.3k	-	Incorrect Format	✅
Gin	Go Web Framework		71.4k	CVE-2023-29401	Double-Quote	✅
Django	Python Web Framework		73k	CVE-2022-36359	Double-Quote	✅
Ktor	Kotlin Web Framework	Caused by insufficient encoding of filename*.	11.5k	CVE-2022-38179	URL Encode	✅
Java (JDK)	Programing Lang		-	CVE-2023-22006	-	✅

Case: Iris

The cause was improper formatting of filename.

Basic format	filename=“abc.txt”
	filename=abc.txt

Iris format	filename=abc.txt

Case: Iris

The cause was improper formatting of filename.

Basic format	filename=“abc.txt”
	filename=abc.txt

Iris format	filename=abc.txt

filename=

Case: Iris

The cause was improper formatting of filename.

Basic format	filename=“abc.txt”
	filename=abc.txt

Iris format	filename=abc.txt

filename= **a.sh; dummy=.txt**

Case: Iris

The cause was improper formatting of filename.

Basic format	filename="abc.txt"
	filename=abc.txt

Iris format	filename=abc.txt

filename=a.sh; **dummy=.txt**

Case: Ktor

The cause is that the Ktor forgot URL Encode in **filename***.

Case: Ktor

The cause is that the Ktor forgot URL Encode in filename*.

filename*　…?

filename*

- **Format available for non-ASCII file names.**
- **If filename and filename* are mixed, filename* takes precedence**

Format (basic pattern):

filename*={ Char-Set }'{Lang}'{ Percent Encoded file name + extension }

Example:

filename*=utf-8”%26%238364%3B.txt

filename*=utf-8’en’%26%238364%3B.txt

Case: Ktor

The cause is that the Ktor forgot URL Encode in **filename***.

Case: Ktor

The cause is that the Ktor forgot URL Encode in **filename***.

- Ktor did not URL encode the file name and did not set the char-set.

Case: Ktor

The cause is that the Ktor forgot URL Encode in filename*.

- Ktor did not URL encode the file name and did not set the char-set.

Ktor format: **filename*={raw_str}**
Correct format: **filename*={char-set}"{URL Encoded filename}.{extension}**

Case: Ktor

The cause is that the Ktor forgot URL Encode in filename*.

- Ktor did not URL encode the file name and did not set the char-set.

Ktor format: filename*={raw_str}
Correct format: filename*={char-set}”{URL Encoded filename}.{extension}

NULL Byte String
▼
"malicious.sh%00'normal.txt

Case: Ktor

The cause is that the Ktor forgot URL Encode in filename*.

- Ktor did not URL encode the file name and did not set the char-set.

Ktor format: **filename*={raw_str}**
Correct format: **filename*={char-set}’’{URL Encoded filename}.{extension}**

''malicious.sh%00'normal.txt

↓

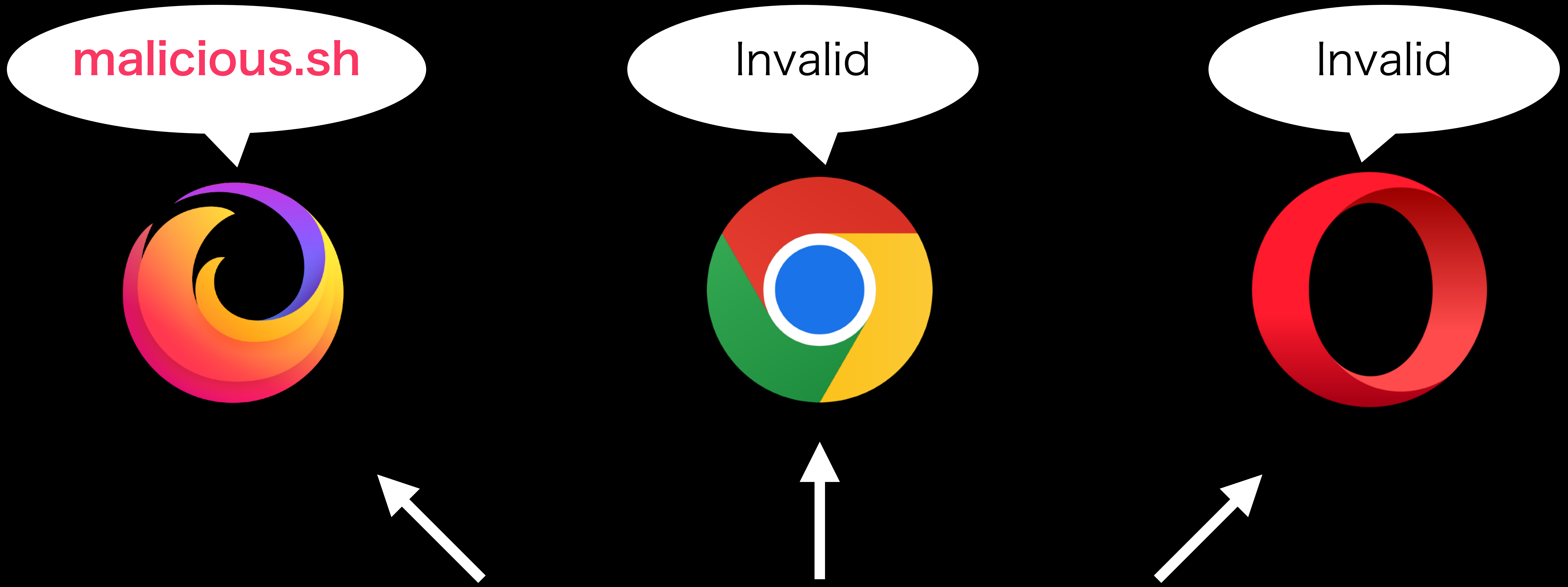
Output:
filename*=''malicious.sh%00'normal.txt

Case: Ktor

Output:

```
filename*=''malicious.sh%00'normal.txt
```

Case: Ktor



malicious.sh

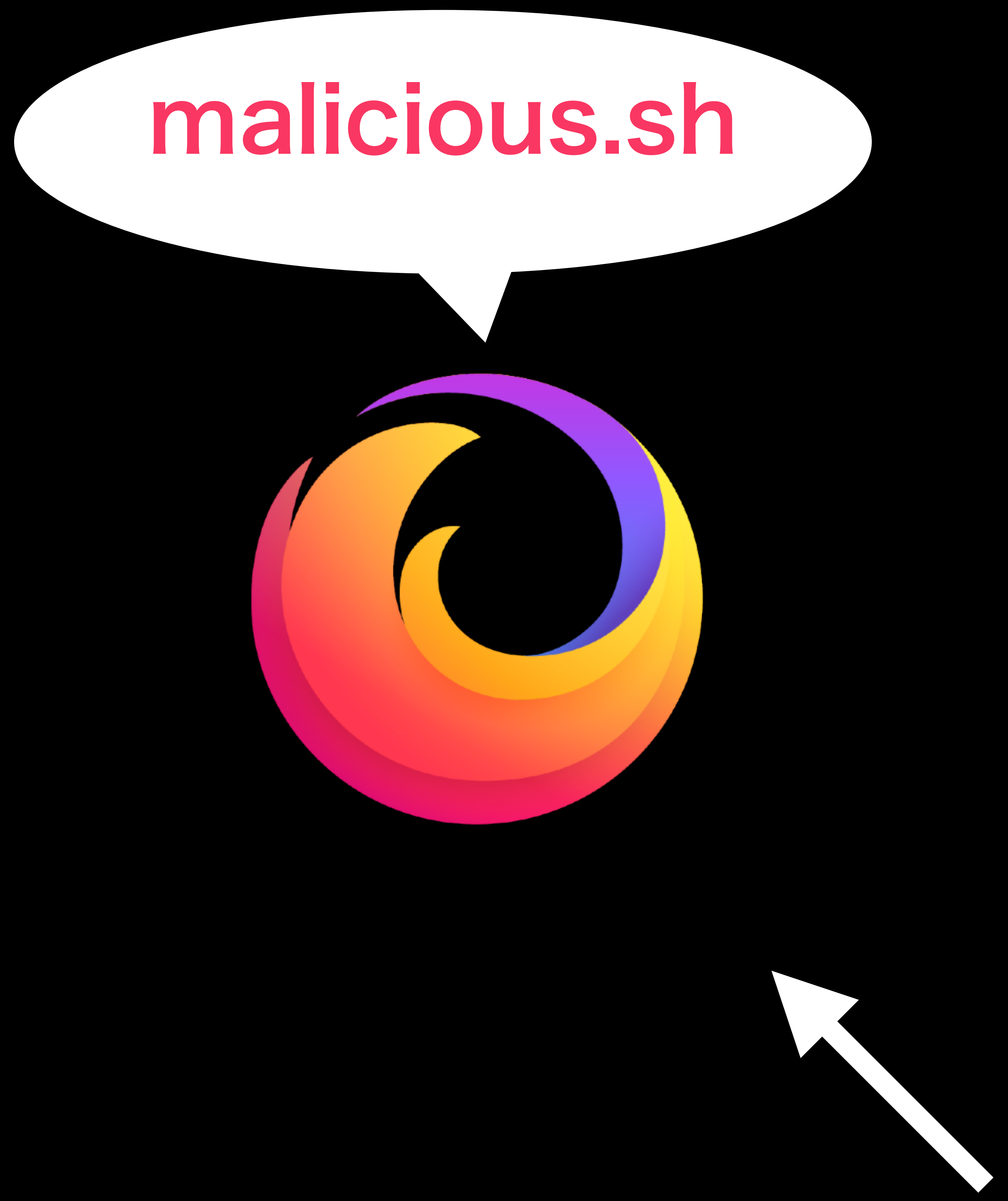
Invalid

Invalid

Output:

filename*=''malicious.sh%00'normal.txt

Case: Ktor



RFD

- Charset does not exists

- Contains NullByte character
  + Single-Quote (‘)

- Single-Quote (‘) , which is the separator between Charset and filename, exists in the middle of the file.

Output:

```
filename*=''malicious.sh%00'normal.txt
```

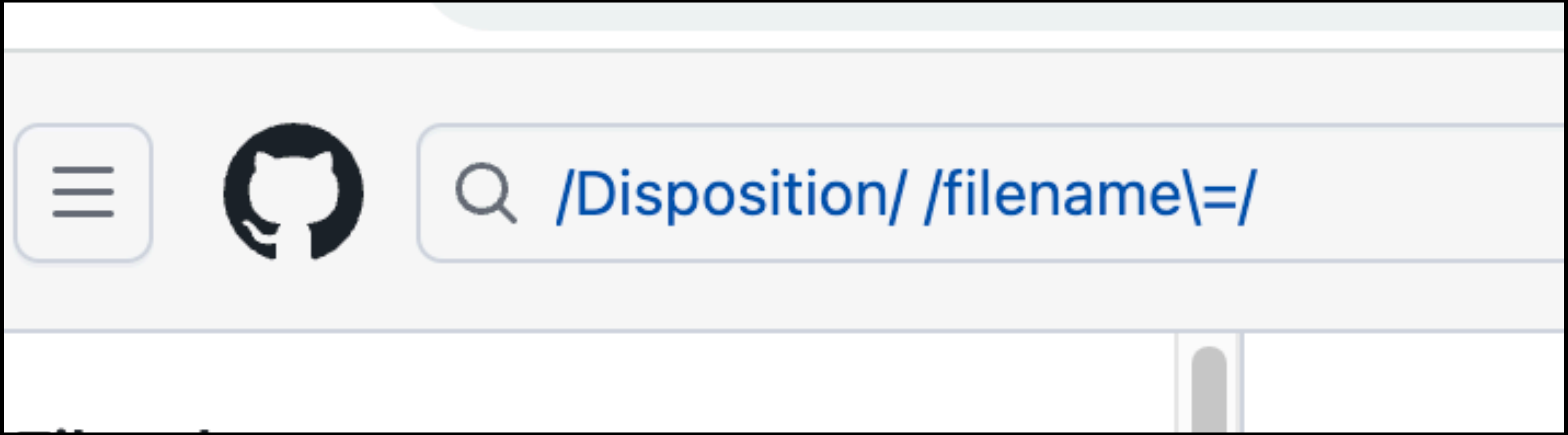
Summary

- ***Content-Dispositon*** have an incomprehensible escaping requirement.
  - **RFC**: “**Unclear**” requirements.
  - **WhatWG HTML Spec**: “**Clear**” requirements.
- The number of systems examine was about 80,
  and many more systems with problems still exist in the world.
- Content-Disposition “**filename**” requires escaping 3 char
  - \r —> %0d or \\r
  - \n —> %0a or \\n
  - “ —> %22 or \”
- Some special cases were not a problem as long as they were RFC compliant.
  (e.g. filename*)

Thank you

Appendix - How to search for Vuln

- https://github.com/search?q=%2FDisposition%2F%20%2Ffilename%5C%3D%2F&type=code
- /Disposition/ /filename=/



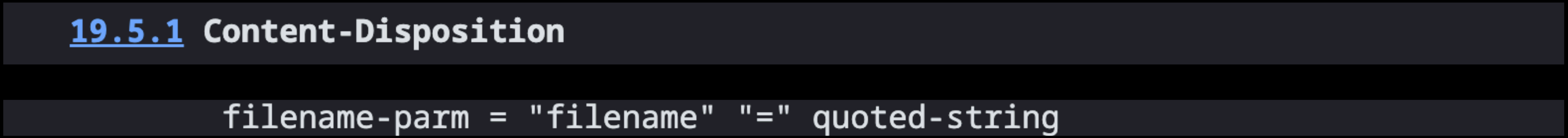
/Disposition/ /filename\=/

Appendix - System and Attack Scenarios

	Request	Email	Response
System Type	Browser HTTP Client	Mail Library	Web Framework
Content-Disposition Output Location	HTTP Body (Multipart)	Mail Body (Multipart)	HTTP Header
Attack Scenarios	Request Tampering Validation Bypass	Request Tampering Validation Bypass	RFD (Reflect File Download)

Appendix - filename="{filename}" format

https://datatracker.ietf.org/doc/html/rfc2616#section-19.5.1 [obsolete]



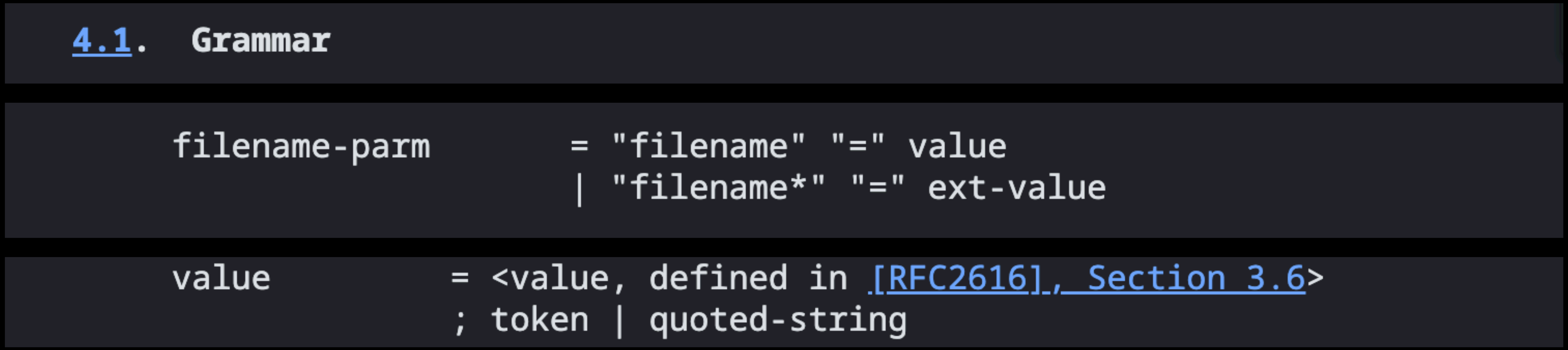
19.5.1 Content-Disposition

```
          filename-parm = "filename" "=" quoted-string
```

filename="abc.txt"

---

https://datatracker.ietf.org/doc/html/rfc6266#section-4.1 [standard]



4.1. Grammar

```
     filename-parm       = "filename" "=" value
                         | "filename*" "=" ext-value

     value           = <value, defined in [RFC2616], Section 3.6>
                     ; token | quoted-string
```

filename=abc.txt or filename="abc.txt"

Reference

- MDN - Content-Disposition
  - https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Disposition

Reference

- RFC
    - https://datatracker.ietf.org/doc/html/rfc2231
    - https://datatracker.ietf.org/doc/html/rfc2616
    - https://datatracker.ietf.org/doc/html/rfc5987
    - https://datatracker.ietf.org/doc/html/rfc6266
    - https://datatracker.ietf.org/doc/html/rfc7578
    - https://datatracker.ietf.org/doc/html/rfc8187

Reference

- WHATWG
  - WHATWG’s html spec > multipart/form-data
    https://html.spec.whatwg.org/multipage/form-control-infrastructure.html#multipart-form-data
  - Issue
    https://github.com/whatwg/html/issues/3223
  - PullRequest:
    https://github.com/whatwg/html/pull/3276
    https://github.com/whatwg/html/pull/6282

Reference

- **Escape example:**
  - **Golang:** https://github.com/golang/go/blob/561a5079057e3a660ab638e1ba957a96c4ff3fd1/src/mime/multipart/writer.go#L132-L136
  - **Symphony:** https://github.com/symfony/symfony/blob/123b1651c4a7e219ba59074441badfac65525efe/src/Symfony/Component/Mime/Header/ParameterizedHeader.php#L128-L133
  - **Spring:** https://github.com/spring-projects/spring-framework/blob/4cc91e46b210b4e4e7ed182f93994511391b54ed/spring-web/src/main/java/org/springframework/http/ContentDisposition.java#L259-L267

Reference

- **Issue:**
  - **Firefox:**
    https://bugzilla.mozilla.org/show_bug.cgi?id=1556711
- **PullRequest:**
  - **Playframework**
    (Detailed research on RFCs and other implementations)
    https://github.com/playframework/playframework/pull/11571

Reference

- Report:
  - My report:
    - English
      https://gist.github.com/motoyasu-saburi/1b19ef18e96776fe90ba1b9f910fa714
    - Japanese:
      https://brutalgoblin.hatenablog.jp/entry/2023/01/05/190150