Theory: HTTP URL

10 minutes 0 / 5 problems solved

§1. What is URL?

Imagine all files on the Internet are located in a megalopolis, each of them lucky to be living in their own home. Given the scale of the Internet, in the resulting settlement, there will be an unimaginable number of blocks and streets. For example, an interesting article, which you want to share with your friend, lives in one of the houses. How do you explain where exactly to find the article? That's right, you need to come up with a single standard by which you could name all the addresses in the city, and then give your friend the street name and house number, just like in a real city!

All documents on the Internet have a personal address. For example, the URL of the JetBrains website looks like this:

```
https://www.jetbrains.com/
```

Web pages, images, videos, and other documents that can be stored on your computer also have addresses. To make them look the same on the Internet, in 1990 the creators of the World Wide Web developed a special standard that defines what addresses should look like. That standard is called a **URL,** which stands for **Uniform Resource Locator**. It represents the standardized way of recording file addresses on the Internet.

The standard has one specific feature: not all characters can be used in URLs. The list of allowed characters includes:

- Latin alphabet (or English alphabet symbols);
- Cyrillic alphabet;
- Numbers;
- Reserved characters with special meaning `!#$&'()*+,/:;=?@[]`;
- Unreserved characters: `-_.~`.

§2. Basic URL Structure

Here is an example of a URL address:

URL address has a certain structure based on the following template:

```
<protocol>://<login>:<password>@<host>:<port>/<path>?<request parameters>#<anchor>
```

Now let's look at this template in more detail:

- `<protocol>` is a way of exchanging data with a resource. You are probably most familiar with HTTP and HTTPS protocols, but there are others;
- `<login>` and `<password>` are prefixes that transmit authentication data for some protocols, if necessary;
- `<host>` is the domain name or IP address where the site is located. **Domain** is the name of the site, its address in a network;
- `<port>` is required for connection within the specified host. The official port for HTTP connections is 80, and the alternative is 8080, but it is possible to use any other ports too. The default setting for HTTPS is 443;

Current topic:

HTTP URL

Topic depends on:

✕ HyperText Transfer Protocol

Topic is required for:

HTTP messages

Tags and attributes

Domains

Links

Launching web server

Table of contents:

↑ HTTP URL

§1. What is URL?

§2. Basic URL Structure

§3. Absolute and relative URLs

§4. Conclusion

Feedback & Comments

- `<path>` indicates the exact address of a particular file or page within a domain;
- `<request parameters>` are parameters transmitted to the server. Depending on request parameters, the site may slightly change its display. For example, it is possible to sort the items of a list in a different order;
- `<anchor>` allows you to connect to a specific part of a web page or document.

This is the general structure of any URL. Most often, when accessing web pages and documents located on a web server, most of the parameters are not mandatory and are set automatically.

When you just want to see a particular page on the Internet with your browser, the URL template looks a lot easier:

```
<protocol>://<host>
```

For example, it can be recorded in a form:

```
https://www.google.com
```

This simplification was created to make life easier for ordinary Internet users, but most programmers need to know the complete template, and now you do.

§3. Absolute and relative URLs

As we know, a URL consists of several parts, and when you're browsing through the same site, some elements of it stay the same. Whichever IDE you want to read about on JetBrains, the protocol and host parts of a URL always match https://www.jetbrains.com. For example, let's look at these links:

- https://www.jetbrains.com/pycharm/ about PyCharm
- https://www.jetbrains.com/go/ about GoLang
- https://www.jetbrains.com/idea/ about IntelliJ IDEA

The new information in each link is its `<path>`. There exists another way to locate resources on the same site by only `<path>?<request parameters>#<anchor>`. The full URL is known as **absolute**, and we call **relative** its shorter counterpart.

> You should remember that it would work only on the same site, while you cannot refer to another site by a relative path. Every time you follow the link with a relative URL, it will expand to absolute, where all parts including everything from protocol to port will match the resource you are using this time.

We know that by absolute URLs we can easily find the resource through the Internet, but why do we need relative paths at all since they will be transformed to absolute anyway? Here are the main reasons for that:

- They are short.
- We can easily move the site to another host.
- They are a little bit faster to retrieve by a browser.

§4. Conclusion

Let's sum up what you have learned about URLs in this topic:

- We can locate any resources on the Internet through a URL.
- Each URL consists of several parts, but some of them are optional.
- We can retrieve resources by an absolute URL and then browse them through relative paths.

Report a typo

518 users liked this theory. 9 didn't like it. What about you?

Start practicing

This content was created over 1 year ago and updated 11 days ago. Share your feedback below in comments to help us improve it!

Comments (10) Hints (0) Useful links (0) Show discussion