Introduction to *Web Application Development* with *Python*

Zhou Fan @ ACM Class 2016

i@evensgn.com

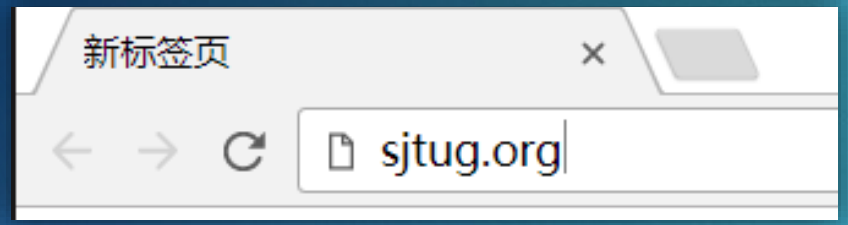
What is a web application

- Web applications
  - Website as front-end
  - eg: Google Docs
- Desktop applications
  - Installed on a local computer
  - eg: Microsoft Word

Advantages of web applications

- Cross platform
- Effective development
- Accessible anywhere
- Easily customisable
- Easier maintenance

HTTP requests and responses



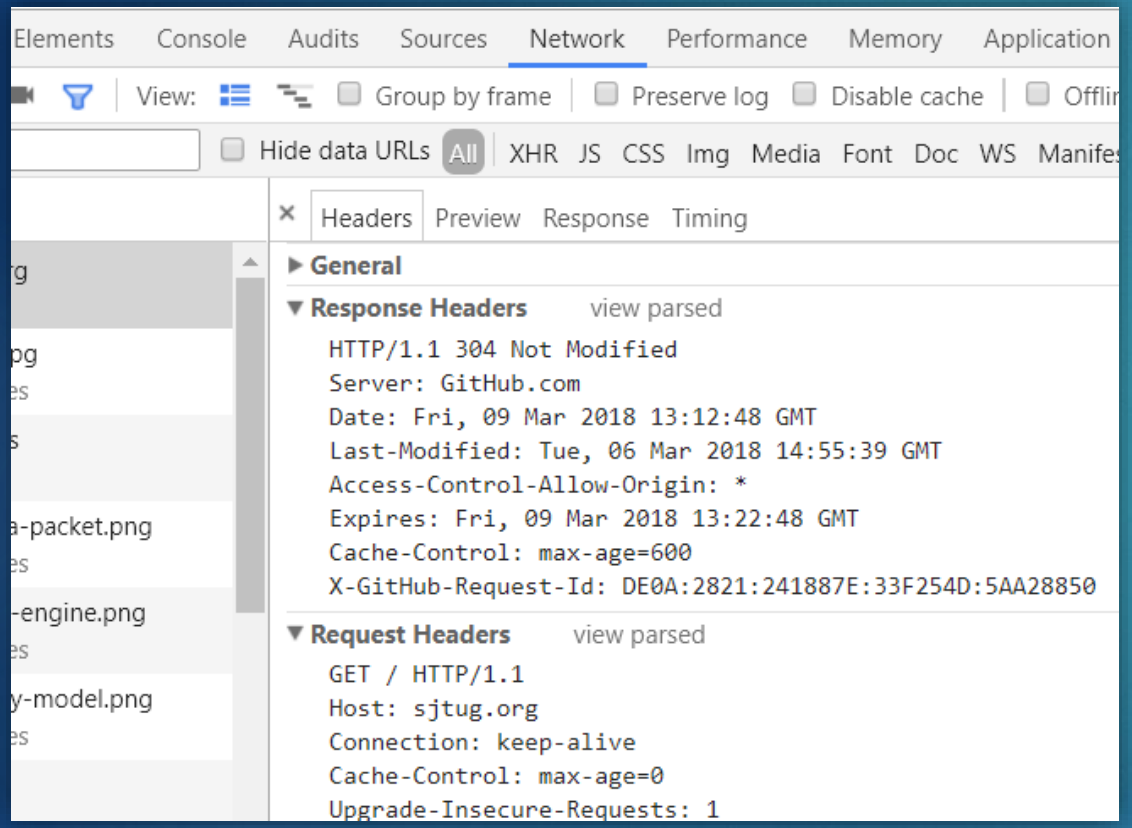
Type a website URL in browser

Get a webpage

HTTP requests and responses

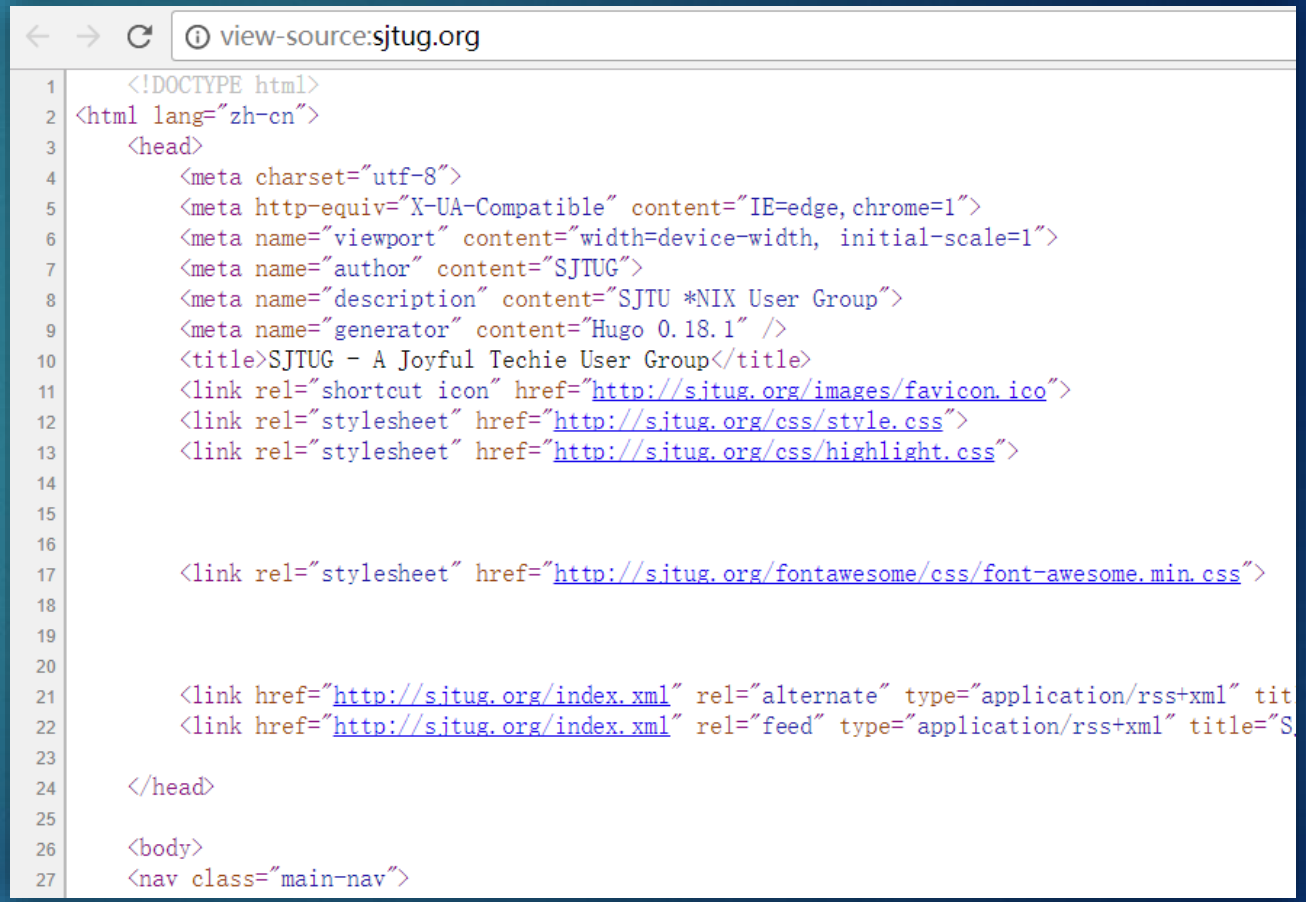
- HTTP: Hypertext Transfer Protocol
- Request Headers
  - `GET / HTTP/1.1`
  - `Host: sjtug.org`
- Response Headers
  - `HTTP/1.1 200 OK`
  - `Content-Type: text/html`

HTTP requests and responses



HTML: Hyper Text Markup Language

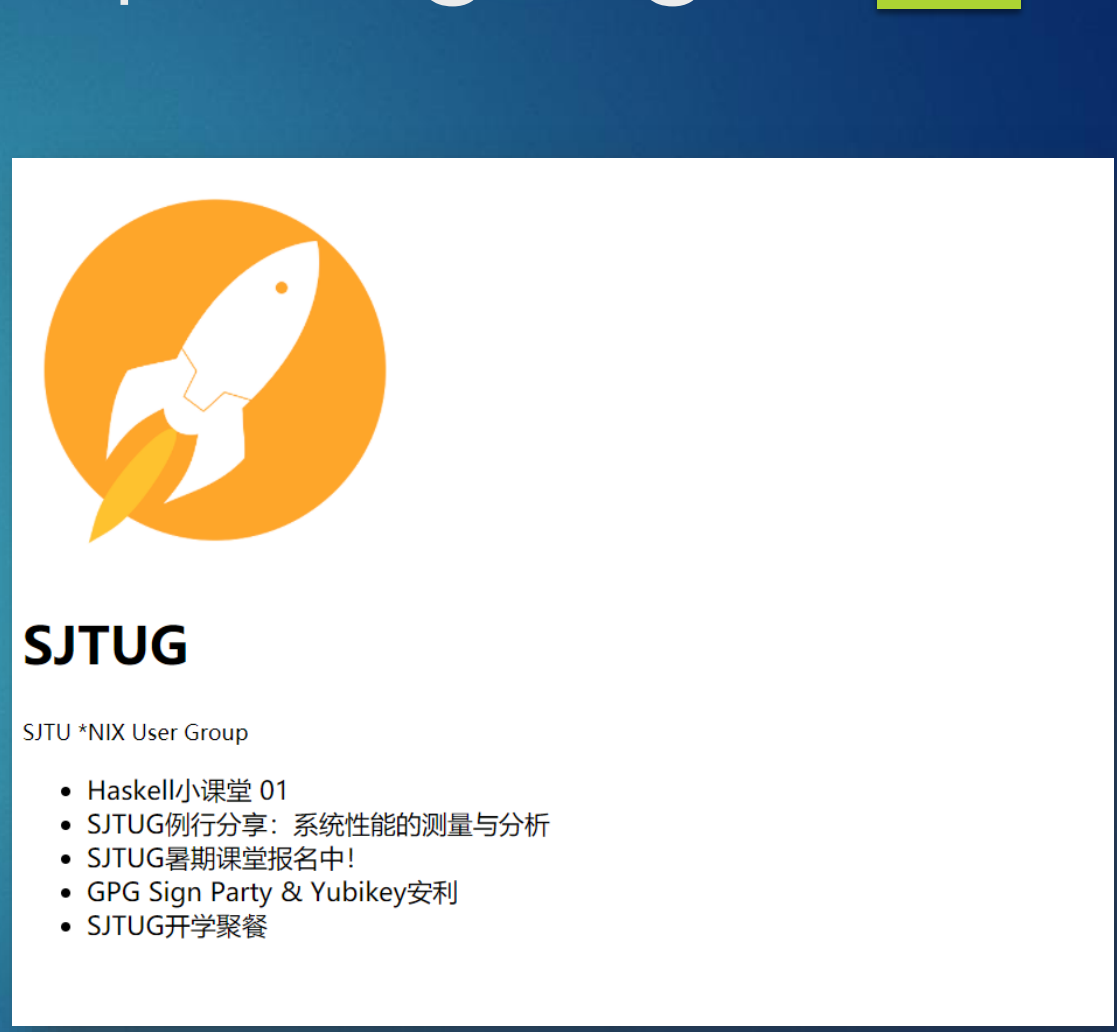
- HTML describes the structure of web pages
- HTML elements are the building blocks of HTML pages
- HTML elements are represented by tags
- Browser use HTML tags to render the web page



HTML: Hyper Text Markup Language

home.html sjtug.css

```html
<!DOCTYPE html>
<html>
    <head>
        <title>SJTUG | Welcome</title>
    </head>
    <body>
        <div class="container">
            <img id="logo" src="./logo.png" width="240">
            <div id="title">
                <h1>SJTUG</h1>
                <small>SJTU *NIX User Group</small>
            </div>
        </div>
        <ul>
            <li>Haskell小课堂 01</li>
            <li>SJTUG例行分享：系统性能的测量与分析</li>
            <li>SJTUG暑期课堂报名中！</li>
            <li>GPG Sign Party &amp;amp; Yubikey安利</li>
            <li>SJTUG开学聚餐</li>
        </ul>
    </body>
</html>
```



CSS: Cascading Style Sheets

- Applies styles selectively to elements in HTML

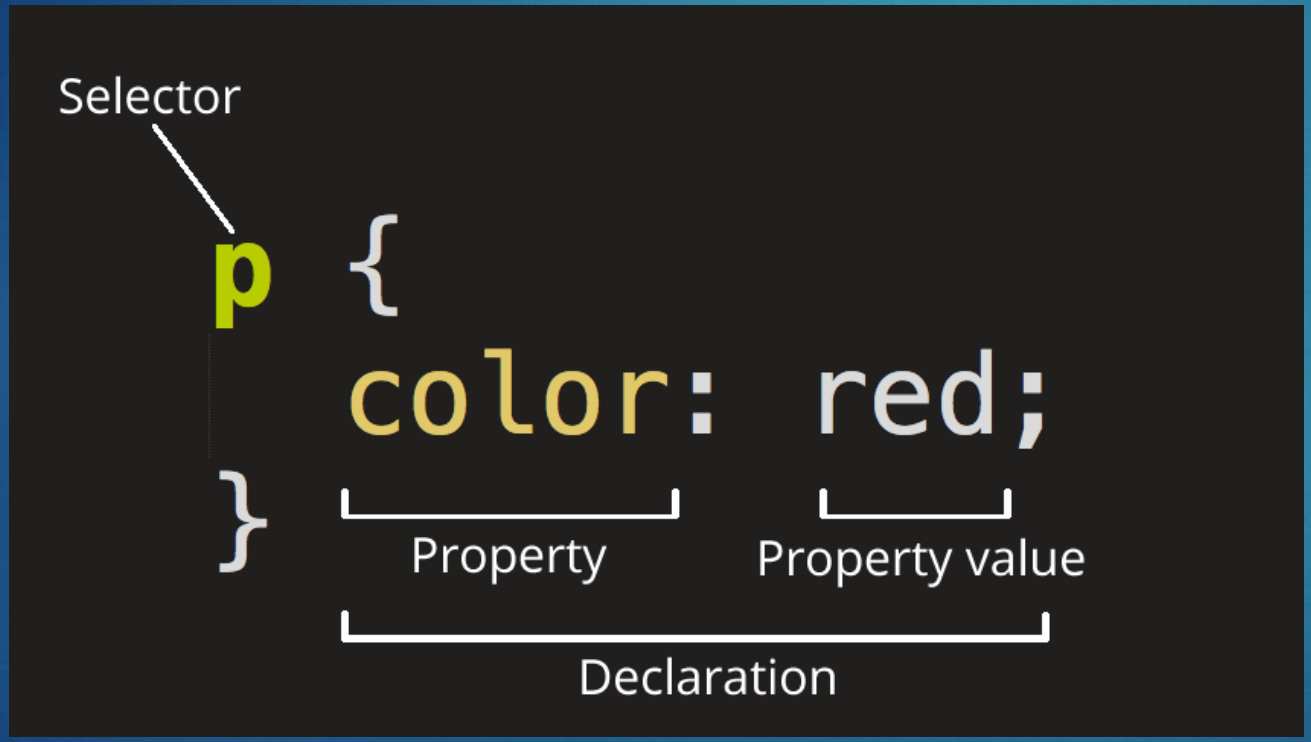


Image from:
https://developer.mozilla.org

CSS: Cascading Style Sheets

home.html | sjtug.css

```
1  .container {
2      width: 240px;
3      margin: 0 auto;
4  }
5
6  #title {
7      text-align: center;
8  }
9
10 #title h1 {
11     font-family: 'Ceviche One', cursive;
12     font-size: 80px;
13     margin: 0;
14 }
15
16 #title small {
17     font-size: 22px;
18     color: #666;
19 }
20
21 @font-face {
22     font-family: 'Ceviche One';
```

SJTUG

SJTU *NIX User Group

- Haskell小课堂 01
- SJTUG例行分享：系统性能的测量与分析
- SJTUG暑期课堂报名中!
- GPG Sign Party & Yubikey安利
- SJTUG开学聚餐

JavaScript

- Provides dynamic interactivity on HTML web pages
- Executed on browsers

```
1 var heading = document.querySelector('h1');
2 heading.textContent = 'Hello';
```

The whole process of a web application

- Browser sends an HTTP request
- Server receives the request and generate an HTML document
- The HTML document is sent to the browser as HTTP response
- The browser receives the HTTP response, then renders the web page using it

WSGI: Web Server Gateway Interface

- a specification for simple and universal interface between web servers and web applications or frameworks for Python

```
1 def application(environ, start_response):
2     start_response('200 OK', [('Content-Type', 'text/html')])
3     return [b'<h1>Hello, web!</h1>']
```

Code from:
https://www.liaoxuefeng.com

WSGI: Web Server Gateway Interface

- Deal with different URLs

```
1 def application(environ, start_response):
2     method = environ['REQUEST_METHOD']
3     path = environ['PATH_INFO']
4     if method=='GET' and path=='/':
5         return handle_home(environ, start_response)
6     if method=='POST' and path='/signin':
7         return handle_signin(environ, start_response)
8     ...
```

Code from:
https://www.liaoxuefeng.com

Flask

- *a microframework for web development in Python*
- *based on Werkzeug, Jinja 2 and good intentions*
- *BSD licensed*

```
@app.route('/', methods=['GET', 'POST'])
def home():
    return '<h1>Welcome to SJTUG</h1>'
```

The templates: Jinja 2

- a modern and designer-friendly templating language for Python


```
<title>{% block title %}{% endblock %}</title>
<ul>
{% for user in users %}
  <li><a href="{{ user.url }}">{{ user.username }}</a></li>
{% endfor %}
</ul>
```

Use SQL in Python: SQLite

```
1  import sqlite3
2  conn = sqlite3.connect('example.db')
3
4  c = conn.cursor()
5  c.execute('''
6            CREATE TABLE person
7            (id INTEGER PRIMARY KEY ASC, name varchar(250) NOT NULL)
8            ''')
9  c.execute('''
10           INSERT INTO person VALUES(1, 'pythoncentral')
11           ''')
12 conn.commit()
13 conn.close()
```

ORM: SQLAlchemy

- ORM: Object-Relational Mapping

```
1 Base = declarative_base()
2
3 class Person(Base):
4     __tablename__ = 'person'
5     id = Column(Integer, primary_key=True)
6     name = Column(String(250), nullable=False)
```

```
1 session = DBSession()
2 # Insert a Person in the person table
3 new_person = Person(name='new person')
4 session.add(new_person)
5 session.commit()
```

Thanks