

WEB GRAPHIC USER INTERFACES

Advanced Programming I

Author: Eng. Carlos Andrés Sierra, M.Sc.
carlos.andres.sierra.v@gmail.com

Computer Engineer
Lecturer
Universidad Distrital Francisco José de Caldas

2024-I



Outline

1 Web Development

2 Model-Template-View (MTV)

3 Web Languages



Outline

1 Web Development

2 Model-Template-View (MTV)

3 Web Languages



What is web development?

- Web development is the process of building, creating, and maintaining websites.
- It includes aspects such as web design, web publishing, web programming, and database management.
- It can range from developing a simple static single page of plain text to complex web-based internet applications (web apps), electronic businesses, and social network services.
- Web development includes many types of web content creation. Some examples include hand coding web pages in a text editor, building a website in a program like Dreamweaver, and updating a blog via a blogging website.



What is web development?

- Web development is the process of **building**, **creating**, and **maintaining** websites.
- It includes aspects such as web design, web publishing, web programming, and database management.
 - *front-end*
 - *backend*
- It can range from developing a simple static single page of main text to complex web-based internet applications (websites), electronic businesses, and social network services.
- Web development includes many types of web content creation. Some examples include hand coding web pages in a text editor, building a website in a program like Dreamweaver, and updating a blog via a blogging website.



What is web development?

- Web development is the process of **building**, **creating**, and **maintaining** websites.
- It includes **aspects** such as **web design**, **web publishing**, **web programming**, and **database management**.
- It can range from developing a simple **static single page** of **plain text** to **complex web-based internet applications** (**web apps**), **electronic businesses**, and **social network services**.

wikipedia

chat

dynamic



What is web development?

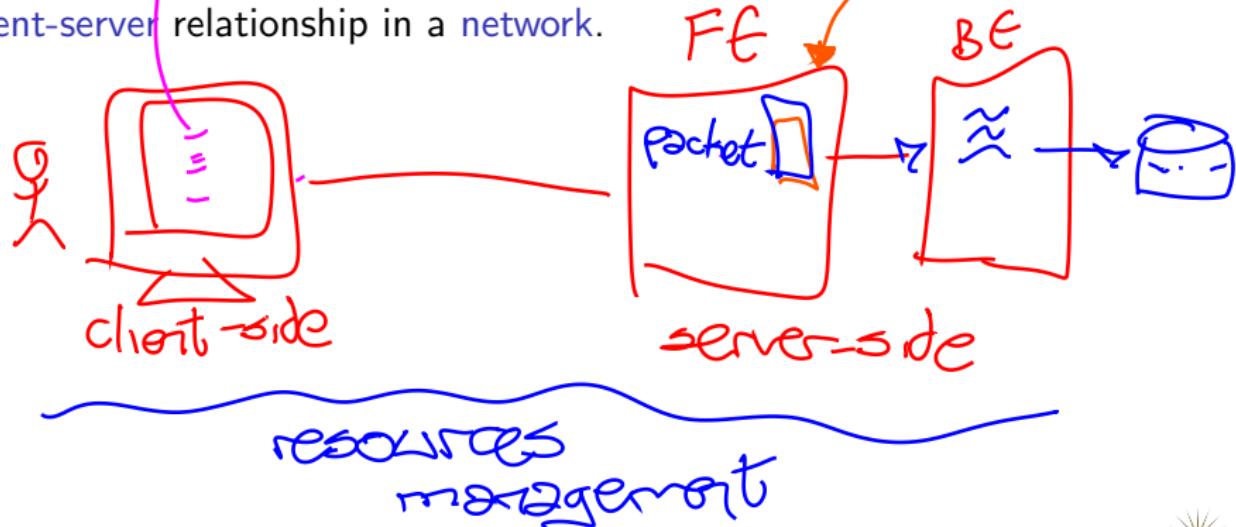
- Web development is the process of **building**, **creating**, and **maintaining** websites.
- It includes **aspects** such as **web design**, **web publishing**, **web programming**, and **database management**.
- It can range from developing a simple **static single page** of plain text to **complex web-based internet applications** (**web apps**), **electronic businesses**, and **social network services**.
- Web development includes many types of web **content creation**. Some examples include **hand coding web pages** in a text editor, **building a website** in a program like **Dreamweaver**, and **updating a blog** via a blogging website.



Client-Side vs. Server-Side

Client-side refers to operations that are performed by the client in a client-server relationship in a network.

form validation \Rightarrow good



Web Browsers

desktop

- A **web browser** is a **software application** for accessing information on the World Wide Web.
- Each individual web page, image, and video is identified by a **distinct URL**.
- A web browser is a client in a **client-server** relationship with a web server.
- The client makes requests to the server and the server sends responses.



Web Browsers

http → www. ~. co

- A **web browser** is a **software application** for accessing information on the **World Wide Web**.
- Each individual **web page**, **image**, and **video** is identified by a **distinct URL**.

Uniform resource Locator

- A web browser is a client in a client-server relationship with a web server.
- The **client** makes requests to the **server** and the **server** sends responses.

ftp ⇒ file transfer protocol



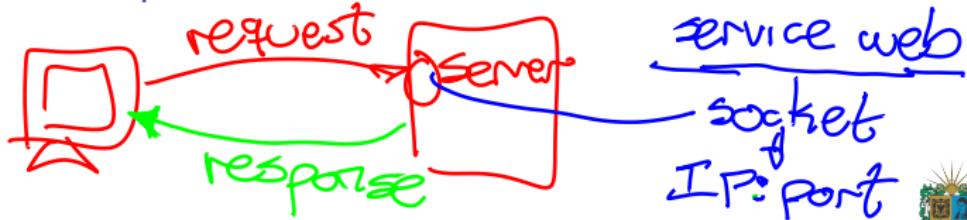
Web Browsers

- A **web browser** is a **software application** for accessing information on the **World Wide Web**.
- Each individual **web page**, **image**, and **video** is identified by a **distinct URL**.
- A **web browser** is a **client** in a **client-server** relationship with a **web server**.
- The client makes requests to the server and the server sends responses.



Web Browsers

- A **web browser** is a **software application** for accessing information on the **World Wide Web**.
- Each individual **web page**, **image**, and **video** is identified by a **distinct URL**.
- A **web browser** is a **client** in a **client-server** relationship with a **web server**.
- The **client** makes **requests** to the **server** and the **server** sends **responses**.



Outline

1 Web Development

2 Model-Template-View (MTV)

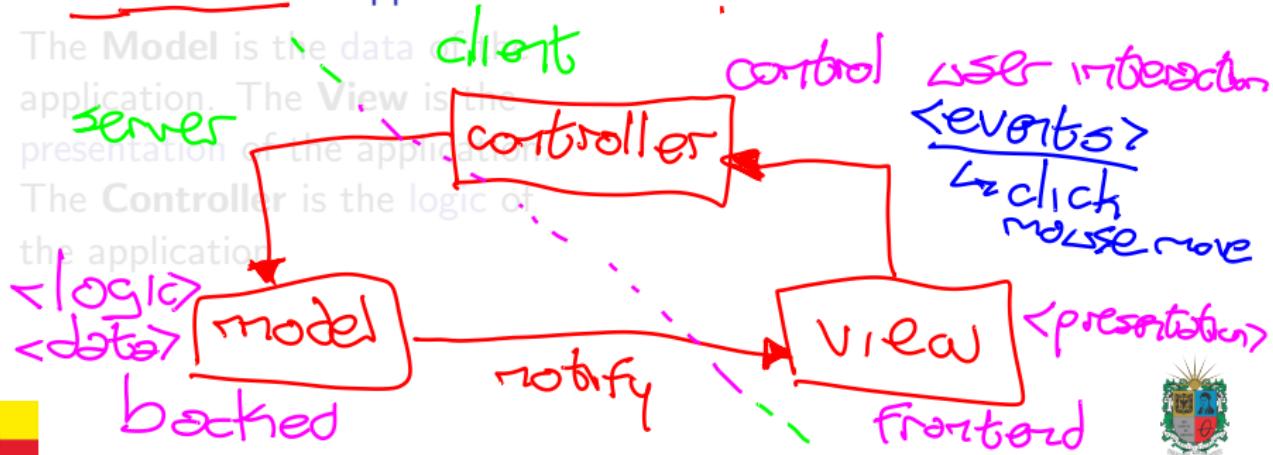
3 Web Languages



Model-View-Controller (MVC)

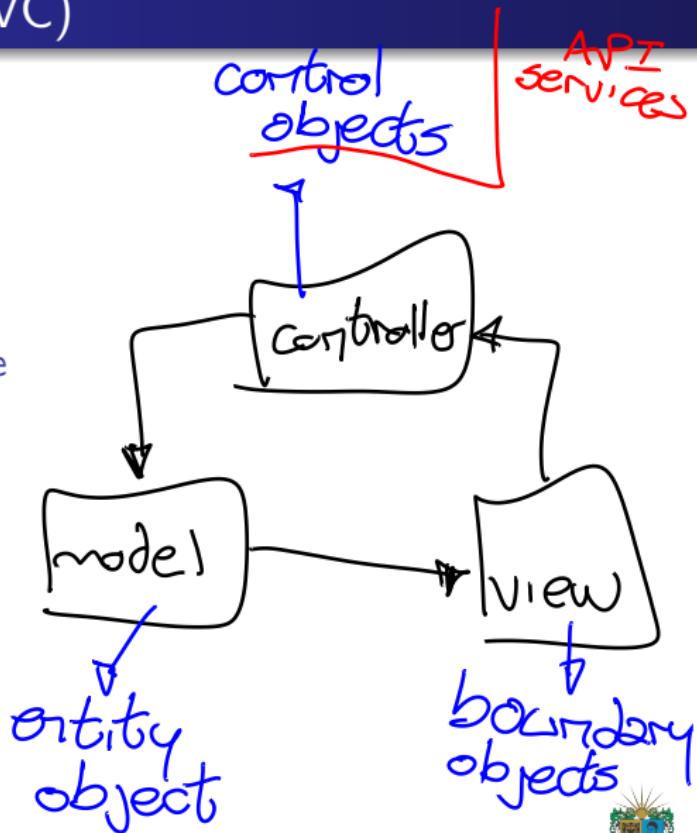
- The **Model-View-Controller (MVC)** is a software design pattern that is used to separate the concerns of an application.

client
The Model is the data of the application. The View is the presentation of the application.
The Controller is the logic of the application.

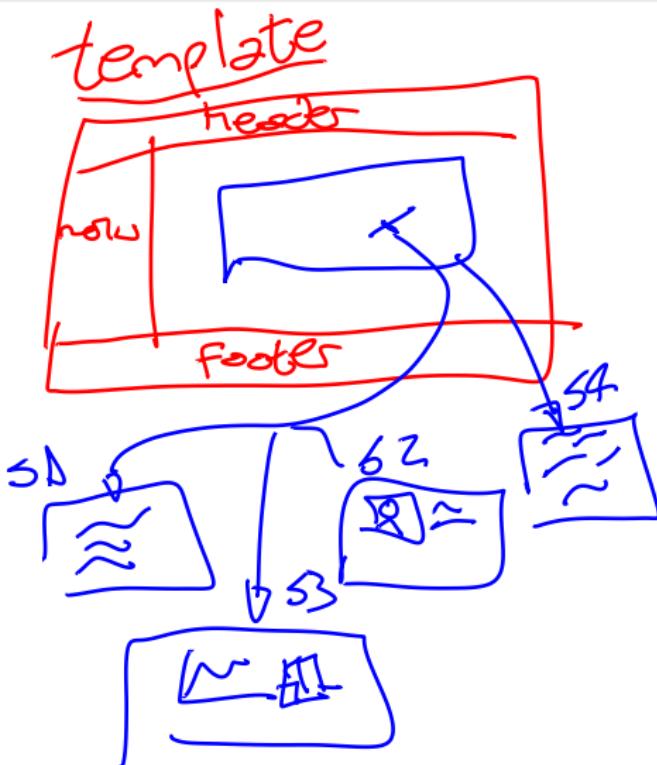


Model-View-Controller (MVC)

- The **Model-View-Controller (MVC)** is a software design pattern that is used to separate the concerns of an application.
- The **Model** is the data of the application. The **View** is the presentation of the application. The **Controller** is the logic of the application.



Model-Template-View (MTV)



- The Model-Template-View (**MTV**) is a software design pattern that is used to separate the concerns of a web application.
 - The **MTV** is a modern way of building web applications. This is because it is simple and easy to understand.
 - The **Model** is the data of the application, the **Template** is the presentation of the application, and the **View** is the logic of the application.

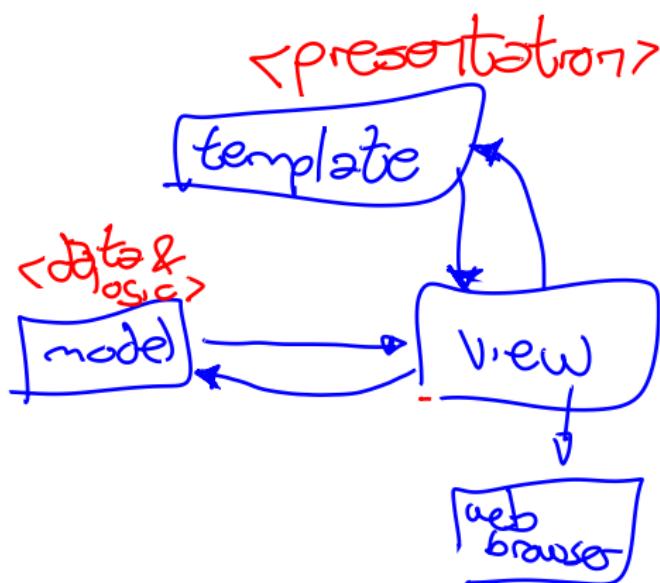


Model-Template-View (MTV)

- The **Model-Template-View (MTV)** is a software design pattern that is used to separate the concerns of a web application.
- The **MTV** is a modern way of building web applications. This is because it is simple and easy to understand.
- The **Model** is the data of the application, the **Template** is the presentation of the application, and the **View** is the logic of the application.



Model-Template-View (MTV)

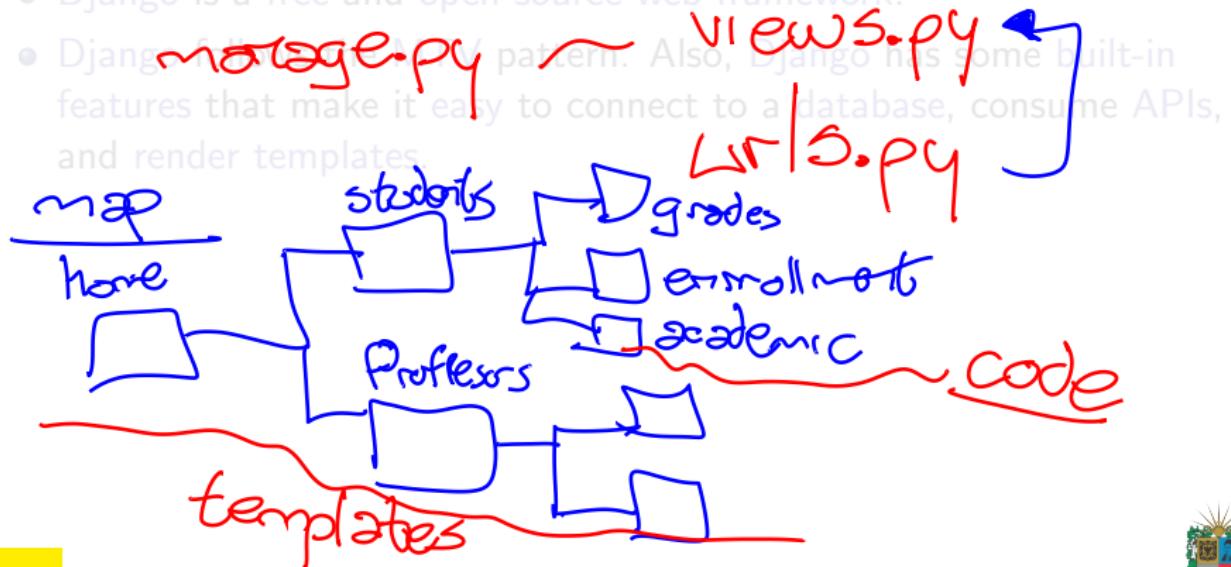


- The **Model-Template-View (MTV)** is a software design pattern that is used to separate the concerns of a web application.
- The **MTV** is a modern way of building web applications. This is because it is simple and easy to understand.
- The **Model** is the data of the application, the **Template** is the presentation of the application, and the **View** is the logic of the application.



Django

- Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design.
- Django is a free and open-source web framework.
- Django follows the MTV pattern. Also, Django has some built-in features that make it easy to connect to a database, consume APIs, and render templates.



Django

- Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design.
- Django is a free and open-source web framework.
- Django follows the MTV pattern. Additionally, Django has some built-in features that make it easy to connect to a database, consume APIs, and render templates.



Django

- Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design.
- Django is a free and open-source web framework.
- Django follows the MTV pattern. Also, Django has some built-in features that make it easy to connect to a database, consume APIs, and render templates.



Outline

1 Web Development

2 Model-Template-View (MTV)

3 Web Languages



HTML

Hypertext Markup Language

- HTML has a typical structure by file: head and body.
 - The head contains the metadata of the document such as the title and links to stylesheets.
 - The body contains the content of the document, such as headings, paragraphs and images. This is based on tags.
 - Next are common HTML tags:
 - h1: Heading 1
 - p: Paragraph
 - img: Image
 - a: Anchor
 - ul: Unordered List
 - li: List Item
 - table: Table, tr: Table Row, td: Table Cell
 - input: Input, button: Button
- `<html>` `<head>` `</head>` `<body>` `</body>` `</html>`
- metadata \Rightarrow browser
- data \Rightarrow user
- template
-



HTML

$\mathfrak{S} \Rightarrow \star$ sturm

- HTML has a typical structure by file: head and body.
 - The head contains the metadata of the document, such as the title and links to stylesheets.
 - The body contains the content of the document, such as headings,

`<head>` contains the content of the document, such as headings, paragraphs, and images. This is based on tags.

- Next are common HTML tags:

- h1: Heading 1
- p: Paragraph
- img: Image
- a: Anchor
- ul: Unordered List
- li: List Item
- table: Table, tr: Table Row, td: Table Data
- form: Form Input, button: Button

<meta charset="UTF-8">

<meta name="keywords" content="WD, PA, Python" >

<meta name="author" content="casanova" >

</head>



HTML

- HTML has a typical structure by file: head and body.
- The head contains the metadata of the document, such as the title and links to stylesheets.
- The body contains the content of the document, such as headings, paragraphs, and images. This is based on tags.
- Next are common HTML tags:

- h1: Heading 1
- p: Paragraph
- img: Image
- a: Anchor
- ul: Unordered List
- li: List Item
- table: Table, tr: Table Row, td: Table Data
- form: Form
- input: Input
- button: Button



HTML

$\langle \triangle \rangle$ $\sim \sqrt{\langle |v| \rangle}$ $\sim \sqrt{\langle |p| \rangle}$

- HTML has a typical structure by file: head and body.
 - The head contains the metadata of the document, such as the title and links to stylesheets.
 - The body contains the content of the document, such as headings, paragraphs, and images. This is based on tags. *<h1> Notes </h1>*
 - Next are common HTML tags:

- **h1:** Heading 1

- p: Paragraph
 - img: Image
 - a: Anchor
 - ul: Unordered List
 - li: List Item
 - table: Table, tr: T
 - form: Form, input

<P> ~~~~~
</p>
List UD

``



CSS

- CSS is a style sheet language used for describing the presentation of a document written in HTML.
*Colors
Shapes
Stress
Layout*
- CSS is designed to enable the separation of presentation from content, including layout, colors, and fonts.
- CSS can be applied to HTML documents in three ways:
 - Inline: Using the style attribute in HTML elements.
 - Internal: Using a style element in the head section.
 - External: Using an external CSS file.
- CSS has a typical structure by file: selector and declaration. Similar to a key-value data structure.
- CSS has selectors that are used to select the HTML elements that you want to style.
 - Element Selector: Selects all elements of a specific type.
 - ID Selector: Selects an element with a specific ID.
 - Class Selector: Selects elements with a specific class.
*Responsive
Bootstrap
(less js)*



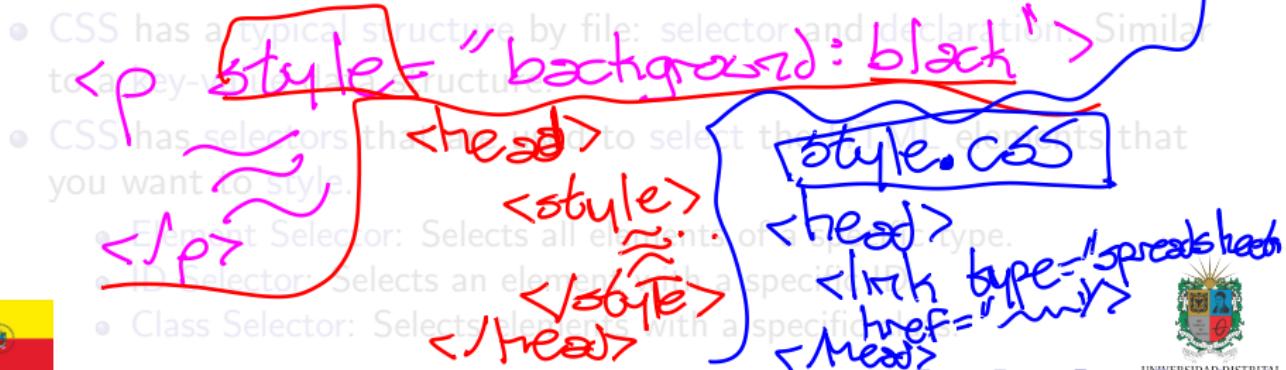
CSS

- CSS is a style sheet language used for describing the presentation of a document written in HTML.
 - CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.
 - CSS can be applied to HTML documents in three ways:
 - Inline: Using the style attribute in HTML elements.
 - Internal: Using a style element in the head section.
 - External: Using an external CSS file.
 - CSS has a typical structure by file: selector and declaration. Similar to a key-value data structure.
 - CSS has selectors that are used to select the HTML elements that you want to style.
 - Element Selector: Selects all elements of a specific type.
 - ID Selector: Selects an element with a specific ID.
 - Class Selector: Selects elements with a specific class.
- Property:* { property: value; }
- selector:* { property: value; }
- declaration:* { property: value; }



CSS

- CSS is a style sheet language used for describing the presentation of a document written in HTML.
- CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.
- CSS can be applied to HTML documents in three ways:
 - Inline: Using the style attribute in HTML elements.
 - Internal: Using a style element in the head section.
 - External: Using an external CSS file.



CSS

- CSS is a style sheet language used for describing the presentation of a document written in HTML.
- CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.
- CSS can be applied to HTML documents in three ways:
 - Inline: Using the style attribute in HTML elements.
 - Internal: Using a style element in the head section.
 - External: Using an external CSS file.
- CSS has a typical structure by file: selector and declaration. Similar to a key-value data structure.
- CSS has selectors that are used to select the HTML elements that you want to style.
 - Element Selector: Selects all elements of a specific type.
 - ID Selector: Selects an element with a specific ID.
 - Class Selector: Selects elements with a specific class.

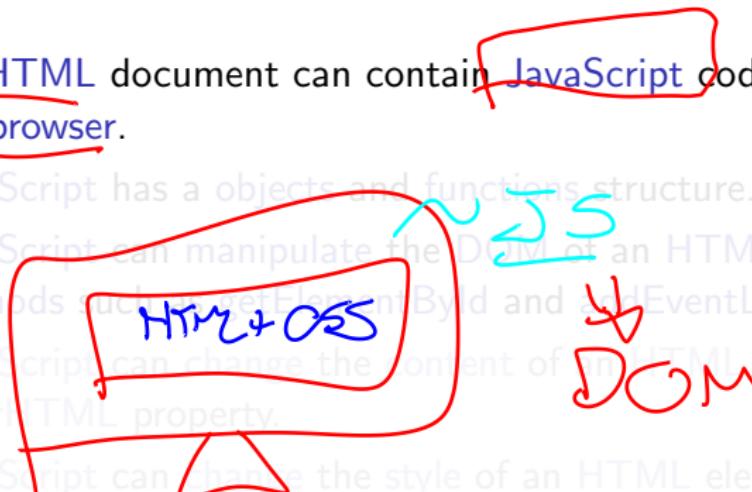


CSS

- CSS is a style sheet language used for describing the presentation of a document written in HTML.
- CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.
- CSS can be applied to HTML documents in three ways:
 - Inline: Using the style attribute in HTML elements.
 - Internal: Using a style element in the head section.
 - External: Using an external CSS file.
- CSS has a typical structure by file: selector and declaration. Similar to a key-value data structure.
- CSS has selectors that are used to select the HTML elements that you want to style.
 - Element Selector: Selects all elements of a specific type.
 - ID Selector: Selects an element with a specific ID.
 - Class Selector: Selects elements with a specific class.



JavaScript

- An HTML document can contain JavaScript code that is executed by the browser.
 - JavaScript has a objects and functions structure.
 - JavaScript can manipulate the DOM of an HTML document using methods such as `getElementsBy` and `addEventListener`.
 - JavaScript can change the content of an HTML element using the `innerHTML` property.
 - JavaScript can change the style of an HTML element using the `style` property.
 - JavaScript can validate forms using the `submit` event.
- 



JavaScript

See x... Let x...

- An **HTML** document can contain **JavaScript** code that is executed by the **browser**.

- JavaScript has a **objects** and **functions** structure.

- JavaScript can manipulate the **DOM** of an **HTML** document using methods such as `getElementsByID` and `getElementsByTagName`.

- JavaScript can change the content of an **HTML** element using the **innerHTML** property.

- JavaScript can change the style of an **HTML** element using the **style** property.

- JavaScript can validate forms using the **submit** event.



Object = key-value

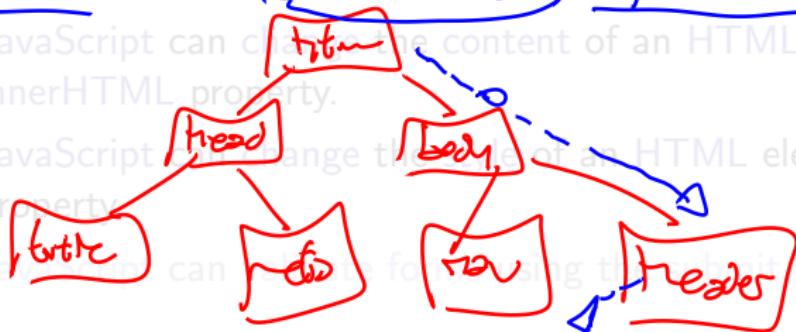


JSON
JavaScript Object Notation



JavaScript

- An **HTML** document can contain **JavaScript** code that is **executed** by the **browser**.
- JavaScript has a **objects** and **functions** structure.
- JavaScript can **manipulate** the **DOM** of an **HTML** document using **methods** such as `getElementByld` and `addEventListener`.
- JavaScript can change the content of an **HTML** element using the `innerHTML` property.
- JavaScript can change the style of an **HTML** element using the `style` property.
- JavaScript can handle events using the `onEvent` event.



JavaScript

- An **HTML** document can contain **JavaScript** code that is **executed** by the **browser**.
- **JavaScript** has a **objects** and **functions** structure.
- **JavaScript** can **manipulate** the **DOM** of an **HTML** document using **methods** such as `getElementById` and `addEventListener`.
- **JavaScript** can **change** the **content** of an **HTML** element using the **innerHTML** property.
- **JavaScript** can change the style of an **HTML** element using the **style** property.
- **JavaScript** can validate forms using the **submit** event.



JavaScript

- An **HTML** document can contain **JavaScript** code that is **executed** by the **browser**.
- **JavaScript** has a **objects** and **functions** structure.
- **JavaScript** can **manipulate** the **DOM** of an **HTML** document using **methods** such as `getElementById` and `addEventListener`.
- **JavaScript** can **change** the **content** of an **HTML** element using the **innerHTML** property.
- **JavaScript** can **change** the **style** of an **HTML** element using the **style** property.
- **JavaScript** can validate forms using the submit event.



JavaScript

- An **HTML** document can contain **JavaScript** code that is **executed** by the **browser**.
- **JavaScript** has a **objects** and **functions** structure.
- **JavaScript** can **manipulate** the **DOM** of an **HTML** document using **methods** such as `getElementById` and `addEventListener`.
- **JavaScript** can **change** the **content** of an **HTML** element using the **innerHTML** property.
- **JavaScript** can **change** the **style** of an **HTML** element using the **style** property.
- **JavaScript** can **validate** forms using the **submit** event.



AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
- AJAX could be used to load content from the server without refreshing the entire page.
- To use AJAX, you need to create an Web API endpoint that returns JSON data.
- AJAX needs to create a request usign the HTTP protocol and headers to specify information for the server.



AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
- AJAX could be used to load content from the server without refreshing the entire page.
- To use AJAX, you need to create an API endpoint that returns JSON data.
- AJAX needs to create a request using the HTTP protocol and headers to specify information for the server.



AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
- AJAX could be used to load content from the server without refreshing the entire page.
- To use AJAX, you need to create an API endpoint that returns JSON data.
- AJAX needs to create a request using the HTTP protocol and headers to specify information for the server.



AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
- AJAX could be used to load content from the server without refreshing the entire page.
- To use AJAX, you need to create an Web API endpoint that returns JSON data.
- AJAX needs to create a request using the HTTP protocol and headers to specify information for the server.



AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
- AJAX could be used to load content from the server without refreshing the entire page.
- To use AJAX, you need to create an Wep API endpoint that returns JSON data.
- AJAX needs to create a request usign the HTTP protocol and headers to specify information for the server.



AJAX

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
- AJAX could be used to load content from the server without refreshing the entire page.
- To use AJAX, you need to create an Wep API endpoint that returns JSON data.
- AJAX needs to create a request usign the HTTP protocol and headers to specify information for the server.



Django

- To install Django, you need to use the pip command.
- To create a Django project, you need to use the django-admin command.
- To create a Django app, you need to use the manage.py command.
- To create a Django model, you need to use the models.Model class.
- To create a Django view, you need to use the views.py file.
- To create a Django template, you need to use the templates folder.
- To create a Django URL, you need to use the urls.py file.
- To create a Django form, you need to use the forms.py file.
- To create a Django admin, you need to use the admin.py file.
- To call an external API in Django, you need to use the requests library.



Django

- To install **Django**, you need to use the **pip** command.
- To create a **Django** project, you need to use the **django-admin** command.
- To create a Django app, you need to use the **manage.py** command.
- To create a Django model, you need to use the **models.Model** class.
- To create a Django view, you need to use the **views.py** file.
- To create a Django template, you need to use the **templates** folder.
- To create a Django URL, you need to use the **urls.py** file.
- To create a Django form, you need to use the **forms.py** file.
- To create a Django admin, you need to use the **admin.py** file.
- To call an external API in Django, you need to use the **requests** library.



Django

- To install **Django**, you need to use the **pip** command.
- To create a **Django** project, you need to use the **django-admin** command.
- To create a **Django** app, you need to use the **manage.py** command.
- To create a **Django** model, you need to use the **models.Model** class.
- To create a **Django** view, you need to use the **views.py** file.
- To create a **Django** template, you need to use the **templates** folder.
- To create a **Django** URL, you need to use the **urls.py** file.
- To create a **Django** form, you need to use the **forms.py** file.
- To create a **Django** admin, you need to use the **admin.py** file.
- To call an external API in **Django**, you need to use the **requests** library.



Django

- To install **Django**, you need to use the `pip` command.
- To create a **Django** project, you need to use the `django-admin` command.
- To create a **Django** app, you need to use the `manage.py` command.
- To create a **Django** model, you need to use the `models.Model` class.
- To create a **Django** view, you need to use the `views.py` file.
- To create a **Django** template, you need to use the `templates` folder.
- To create a **Django** URL, you need to use the `urls.py` file.
- To create a **Django** form, you need to use the `forms.py` file.
- To create a **Django** admin, you need to use the `admin.py` file.
- To call an external API in **Django**, you need to use the `requests` library.



Django

- To install Django, you need to use the pip command.
- To create a Django project, you need to use the django-admin command.
- To create a Django app, you need to use the manage.py command.
- To create a Django model, you need to use the models.Model class.
- To create a Django view, you need to use the views.py file.
- To create a Django template, you need to use the templates folder.
- To create a Django URL, you need to use the urls.py file.
- To create a Django form, you need to use the forms.py file.
- To create a Django admin, you need to use the admin.py file.
- To call an external API in Django, you need to use the requests library.



Django

- To install **Django**, you need to use the **pip** command.
- To create a **Django** project, you need to use the **django-admin** command.
- To create a **Django** app, you need to use the **manage.py** command.
- To create a **Django** model, you need to use the **models.Model** class.
- To create a **Django** view, you need to use the **views.py** file.
- To create a **Django** template, you need to use the **templates** folder.
- To create a **Django** URL, you need to use the **urls.py** file.
- To create a **Django** form, you need to use the **forms.py** file.
- To create a **Django** admin, you need to use the **admin.py** file.
- To call an external API in **Django**, you need to use the **requests** library.



Django

- To install Django, you need to use the pip command.
- To create a Django project, you need to use the django-admin command.
- To create a Django app, you need to use the manage.py command.
- To create a Django model, you need to use the models.Model class.
- To create a Django view, you need to use the views.py file.
- To create a Django template, you need to use the templates folder.
- To create a Django URL, you need to use the urls.py file.
- To create a Django form, you need to use the forms.py file.
- To create a Django admin, you need to use the admin.py file.
- To call an external API in Django, you need to use the requests library.



Django

- To install Django, you need to use the pip command.
- To create a Django project, you need to use the django-admin command.
- To create a Django app, you need to use the manage.py command.
- To create a Django model, you need to use the models.Model class.
- To create a Django view, you need to use the views.py file.
- To create a Django template, you need to use the templates folder.
- To create a Django URL, you need to use the urls.py file.
- To create a Django form, you need to use the forms.py file.
- To create a Django admin, you need to use the admin.py file.
- To call an external API in Django, you need to use the requests library.



Django

- To install Django, you need to use the pip command.
- To create a Django project, you need to use the django-admin command.
- To create a Django app, you need to use the manage.py command.
- To create a Django model, you need to use the models.Model class.
- To create a Django view, you need to use the views.py file.
- To create a Django template, you need to use the templates folder.
- To create a Django URL, you need to use the urls.py file.
- To create a Django form, you need to use the forms.py file.
- To create a Django admin, you need to use the admin.py file.
- To call an external API in Django, you need to use the requests library.



Django

- To install **Django**, you need to use the `pip` command.
- To create a **Django** project, you need to use the `django-admin` command.
- To create a **Django** app, you need to use the `manage.py` command.
- To create a **Django** model, you need to use the `models.Model` class.
- To create a **Django** view, you need to use the `views.py` file.
- To create a **Django** template, you need to use the `templates` folder.
- To create a **Django** URL, you need to use the `urls.py` file.
- To create a **Django** form, you need to use the `forms.py` file.
- To create a **Django** admin, you need to use the `admin.py` file.
- To call an external **API** in **Django**, you need to use the `requests` library.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
 - Template tags are used to display dynamic content in the template.
 - Template filters are used to format data in the template.
 - Django templates can extend other templates using the extends tag.
 - Django templates can include other templates using the include tag.
 - Django templates can loop over data using the for tag.
 - Django templates can conditional display data using the if tag.
 - Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
 - Template filters are used to format data in the template.
 - Django templates can extend other templates using the extends tag.
 - Django templates can include other templates using the include tag.
 - Django templates can loop over data using the for tag.
 - Django templates can conditional display data using the if tag.
 - Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Django & Templates

- Django uses a template system to separate the design from the Python code.
- Django templates are HTML files that contain template tags and filters.
- Template tags are used to display dynamic content in the template.
- Template filters are used to format data in the template.
- Django templates can extend other templates using the extends tag.
- Django templates can include other templates using the include tag.
- Django templates can loop over data using the for tag.
- Django templates can conditional display data using the if tag.
- Django has the static tag to load static files such as images, CSS and JavaScript.



Outline

1 Web Development

2 Model-Template-View (MTV)

3 Web Languages



Thanks!

Questions?



Repo:

 github.com/engandres/ud-public/courses/advanced-programming

