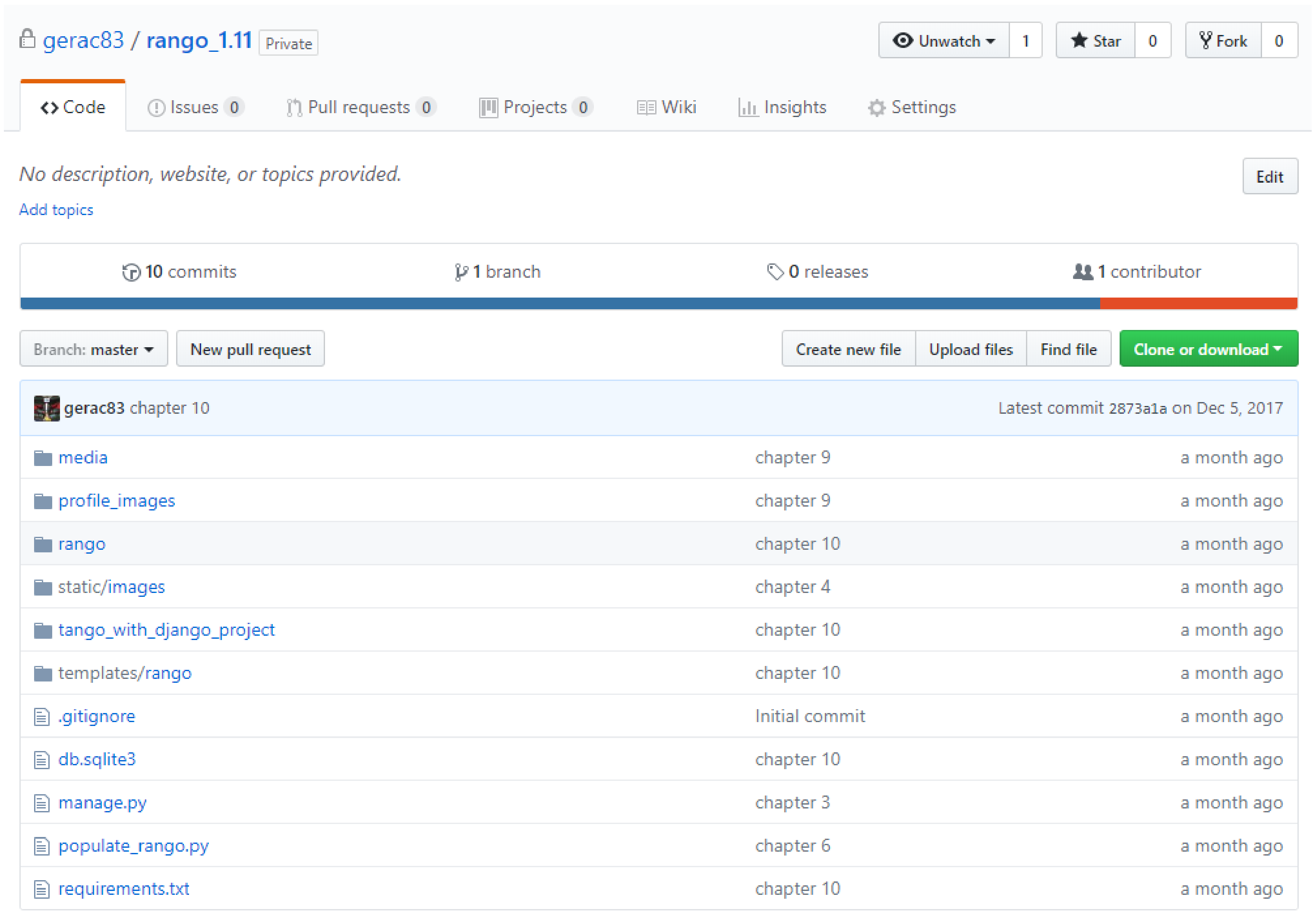
University of Glasgow

Django
Web Application Framework

Internet Technologies

ITECH

GitHub repository for Rango



gerac83 / rango_1.11 Private

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

No description, website, or topics provided. Edit

Add topics

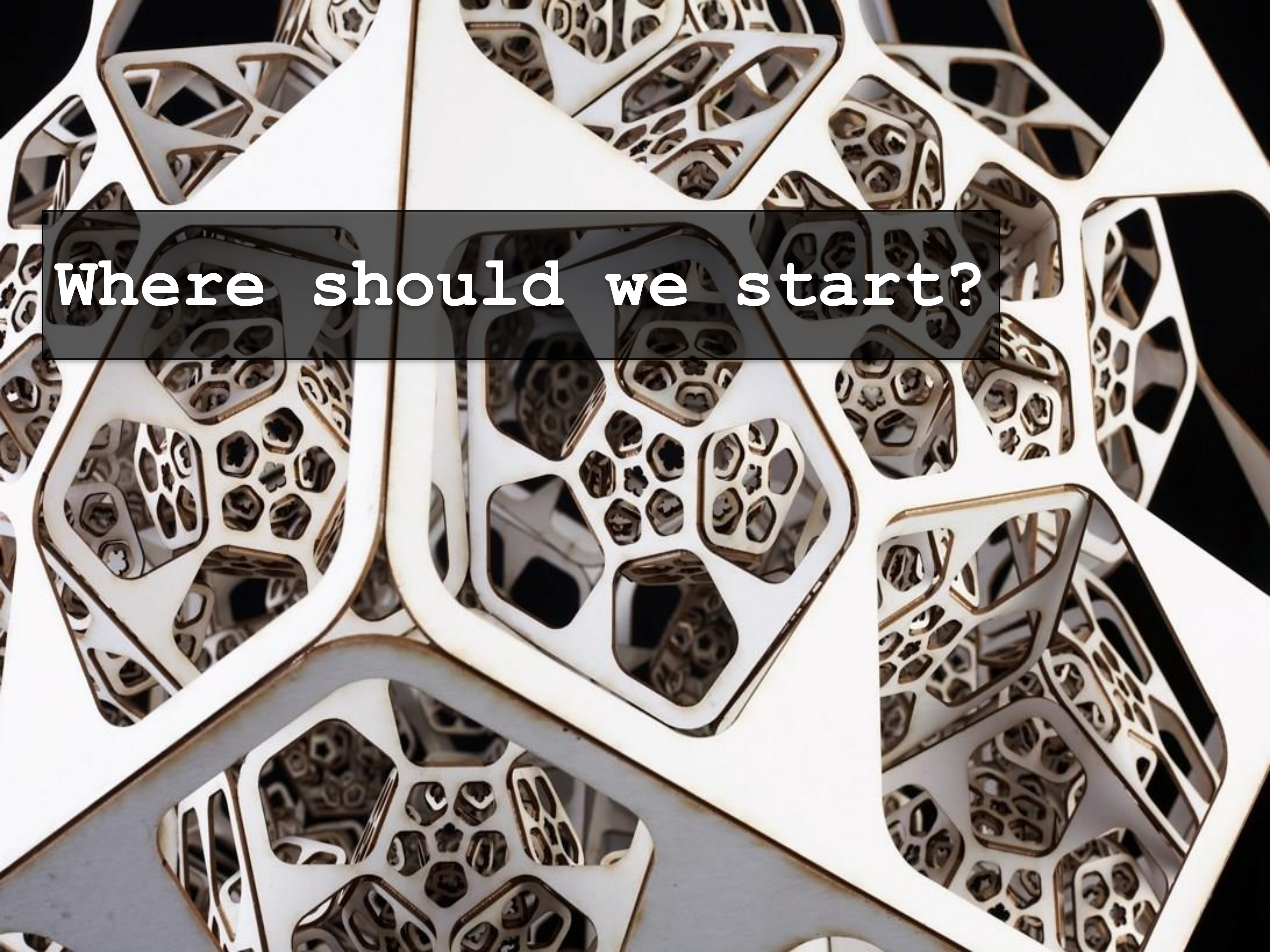
10 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

gerac83 chapter 10 — Latest commit 2873a1a on Dec 5, 2017

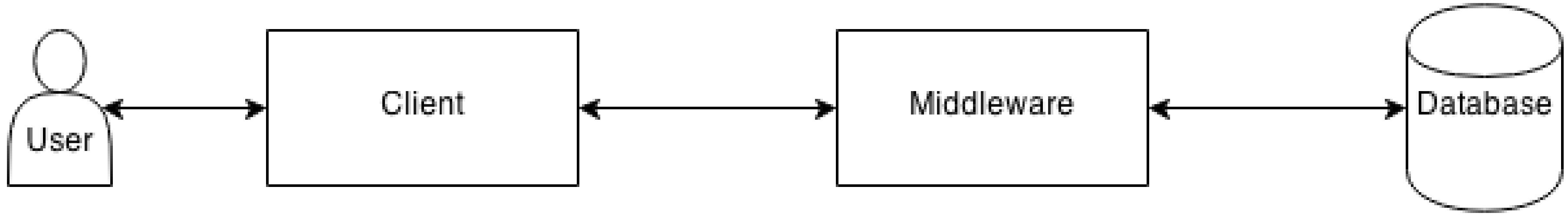
media	chapter 9	a month ago
profile_images	chapter 9	a month ago
rango	chapter 10	a month ago
static/images	chapter 4	a month ago
tango_with_django_project	chapter 10	a month ago
templates/rango	chapter 10	a month ago
.gitignore	Initial commit	a month ago
db.sqlite3	chapter 10	a month ago
manage.py	chapter 3	a month ago
populate_rango.py	chapter 6	a month ago
requirements.txt	chapter 10	a month ago

As web architects we need to see through the complexities, abstract away, and design a useful solution.



Where should we start?

High Level System Architecture



- We need to work out what we are going to build
- And we need to decide what technologies will be used in each box.
- For the middleware, we will be using Django as the WAF for building the application server.

Top-Down vs. Bottom Up System Architecture

Top-Down

Separates the low level work from the higher level abstractions

Leads to a modular design

Development can be self-contained (tiered)

Emphasizes planning and system understanding

Coding is late, and
Testing is even later

Skeleton code can show how everything integrates

Bottom-Up

Coding begins early and so
Testing can be performed early

Requires really good intuitions to determine functionality of modules

Low level design decisions can have major impact on solutions

Risks integration problems – how do components link together

Often used to add on to existing modules

django

Django

- Pronounced JANG-oh

- “*Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design*”

- Claimed to be the web framework for perfectionists with deadlines – you may differ!

Django

- History:
  - Created by Adrian Holovaty and Jacob Kaplan-Moss at World Online news for efficient development
  - Open Sourced in 2005, first major release in 2008
- Primary Focus:
  - Dynamic and database driven websites
  - Content based websites
  - Examples:
    - eBay, Amazon, GumTree, etc
    - The Gaurdian, Herald, etc
    - Match, Twitter, etc

Why Django for Web Dev

- Lets you divide code modules into logical groups
  - Providing flexibility and easier to change
  - Underpinned by the MVC/MVT design pattern
- Provides automatically generated web administration
  - Easier to manage the database
- Provides many pre-packaged APIs for common tasks

Why Django for Web Dev

- Provides a template system to define HTML templates
  - Avoids code duplications
  - Subscribes to Don’t-Repeat-Yourself (DRY) principle
- Allows you to define what the URL will be for a given view
  - Loosely Couple Principle
- Allows you to separate business logic from the presentation
  - Separation of Concerns

Overall Design Philosophy

- **Loose Coupling**
- Less Code
- Quick Development
- **Don’t Repeat yourself (DRY)**
- **Explicit is better than implicit**
  - A core Python principle
- Consistency
- See http://docs.djangoproject.com/en/dev/misc/design-philosophies/ for more details and more philosophies

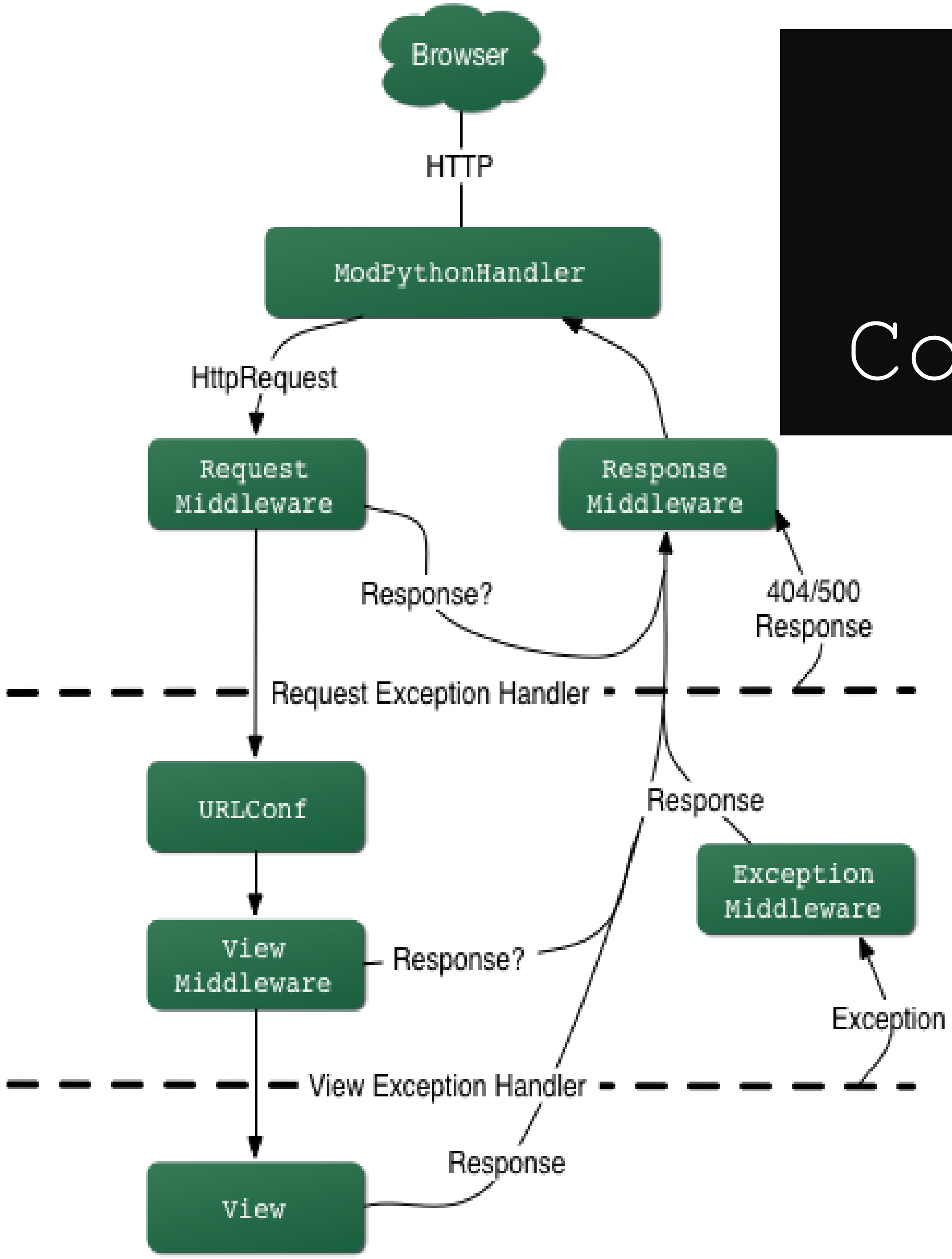
Django Modules

- Administration Interface (CRUD)
  - Create, Read, Update and Delete
- Authentication Systems
- Forms Handling
- Session Handling
- Syndication Frameworks
  - RSS and Atom feeds
- Caching
- Internationalization and Localization
- And much, much more…

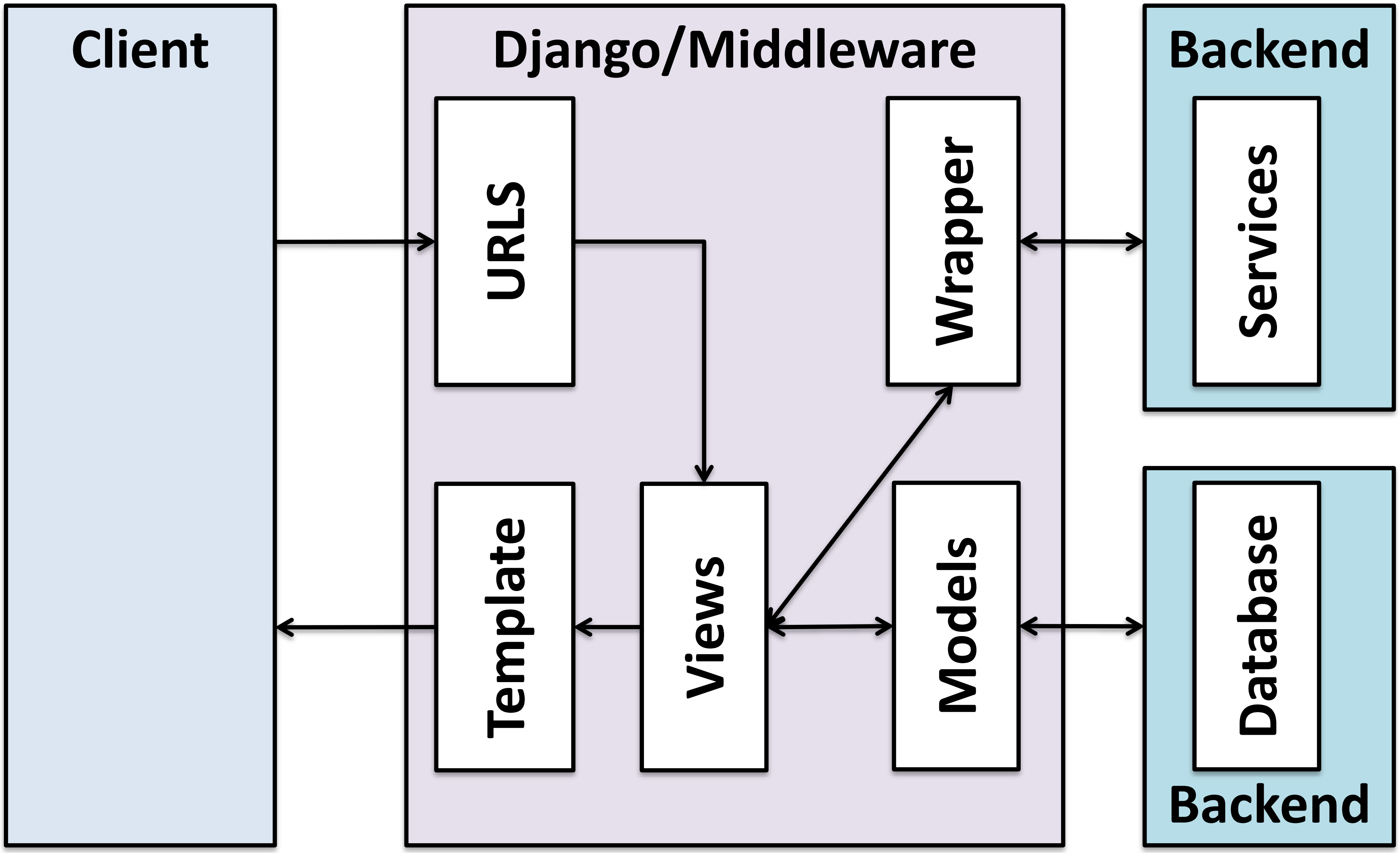
Model View Controller in Django

- **Models** describe your database
- **Views** determines what the user sees
- **Controller** is handled by
  - the Django Framework
  - URL parser maps urls to views
- **Templates** describe how the data is presented
  - Additional to the MVC, i.e., MVCT

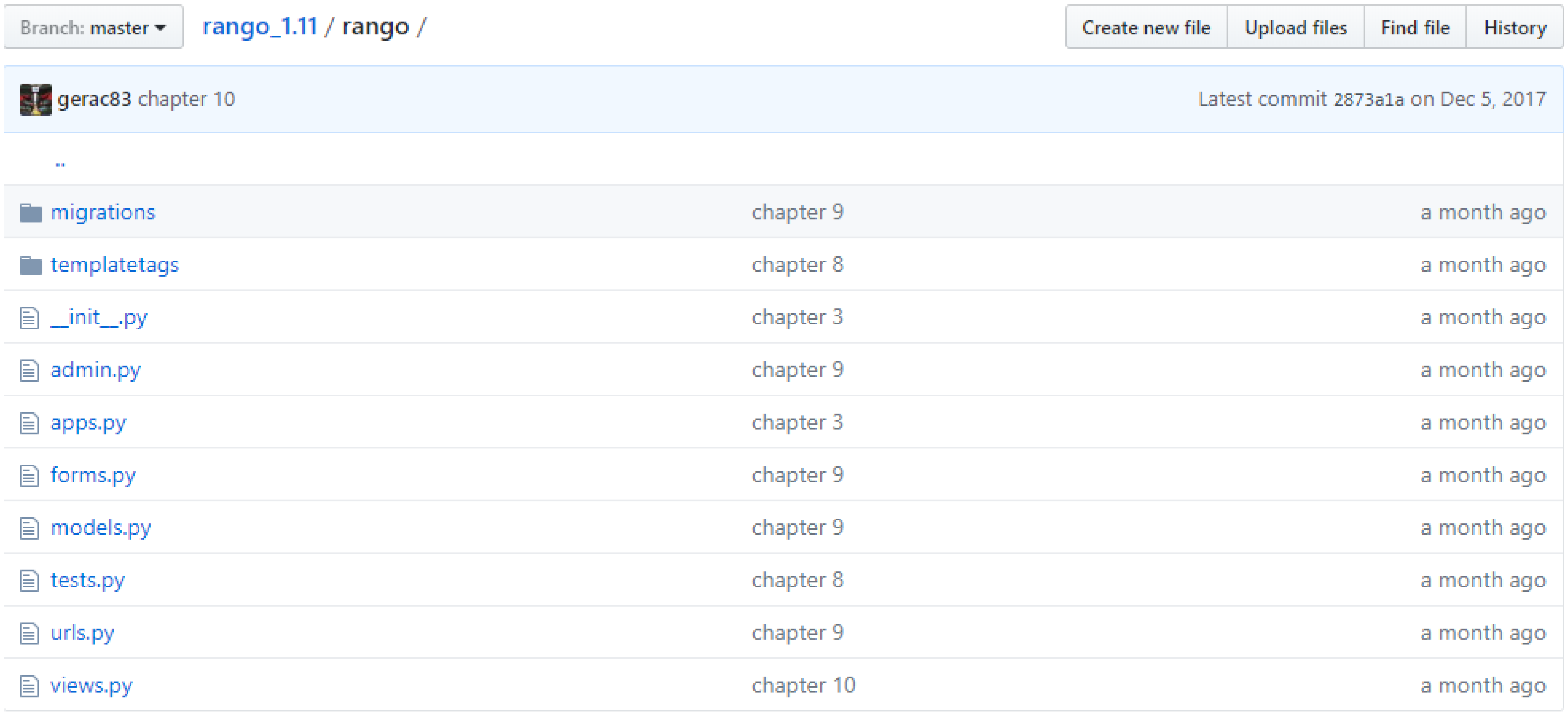
Django Internal Control Flow



Simplified Internal Flow



Internal Sections/Components



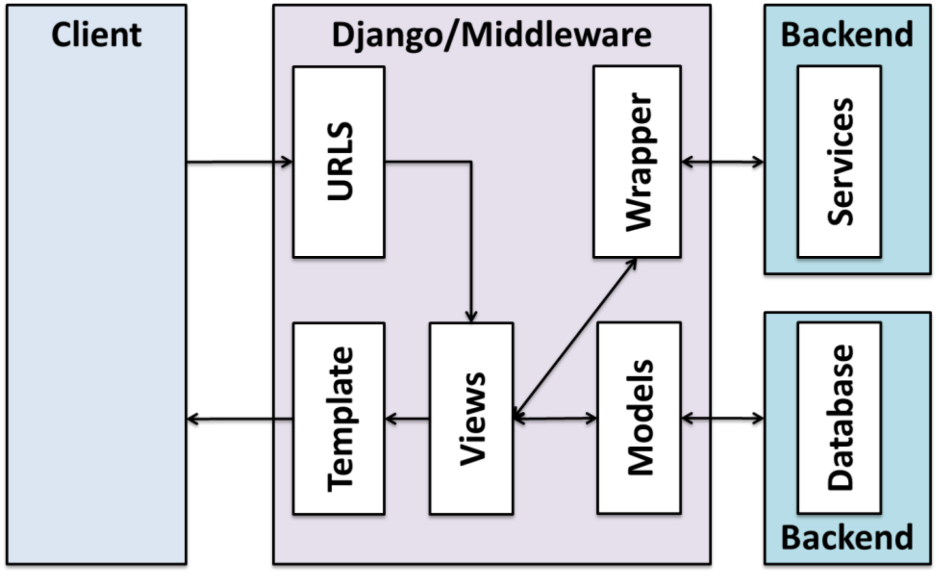
rango_1.11 / rango /

gerac83 chapter 10 | Latest commit 2873a1a on Dec 5, 2017

..

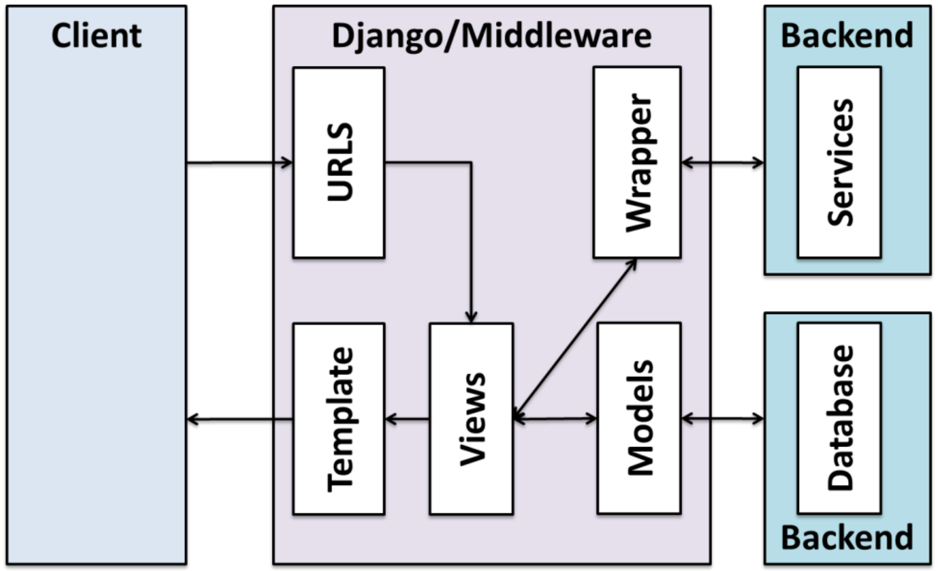
migrations	chapter 9	a month ago
templatetags	chapter 8	a month ago
__init__.py	chapter 3	a month ago
admin.py	chapter 9	a month ago
apps.py	chapter 3	a month ago
forms.py	chapter 9	a month ago
models.py	chapter 9	a month ago
tests.py	chapter 8	a month ago
urls.py	chapter 9	a month ago
views.py	chapter 10	a month ago

Internal Sections/Components



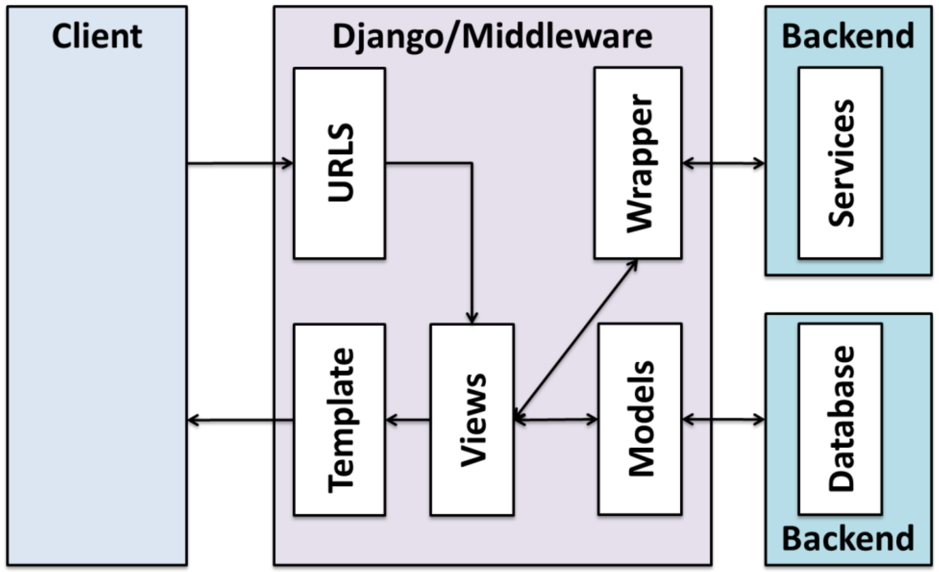
- **Building Data Models (usually in models.py):**
  - The models specify the entities and relationships in the database – these provide an Object Relational Mapping to the actual database tables
  - Django constructs the database given the models defined

Internal Sections/Components



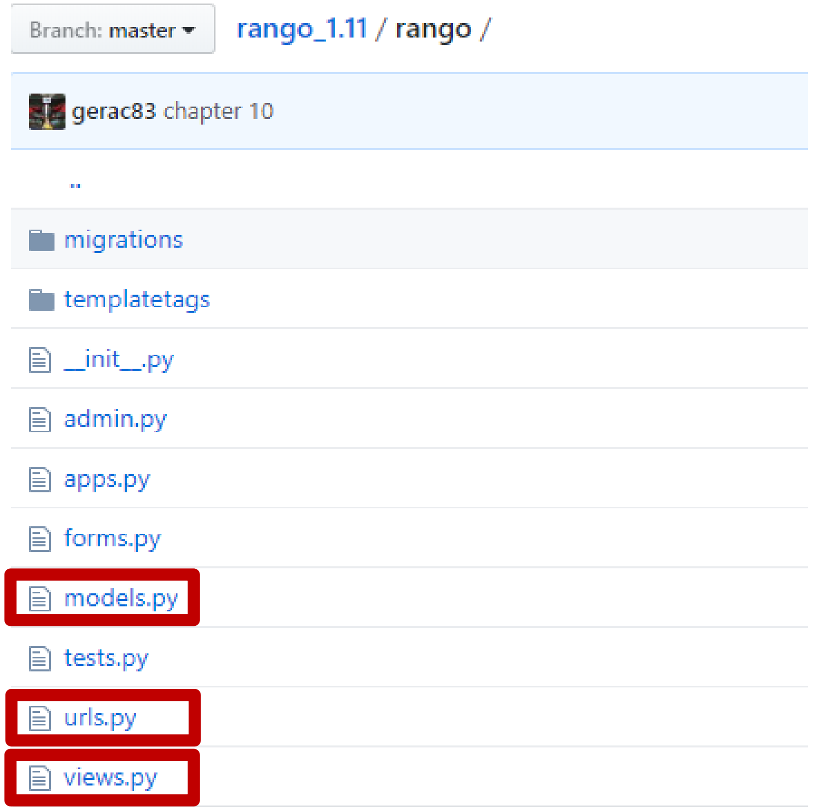
- **Defining Views (usually in views.py):**
  – Views are responsible for handling and processing the specific request, collating the data from databases/external services, then selecting the template, for the response to be generated

Internal Sections/Components



- **Controlling flow (usually in urls.py):**
  - To specify what view function should handled a particular URL (or part of), URL patterns are used to find matches with the URL, and to route this request to the appropriate view
  - The use of pattern matching means that different instantiations can be handled by a common pattern

Internal Sections/Components



Branch: master

rango_1.11 / rango /

gerac83 chapter 10

..

migrations

templatetags

__init__.py

admin.py

apps.py

forms.py

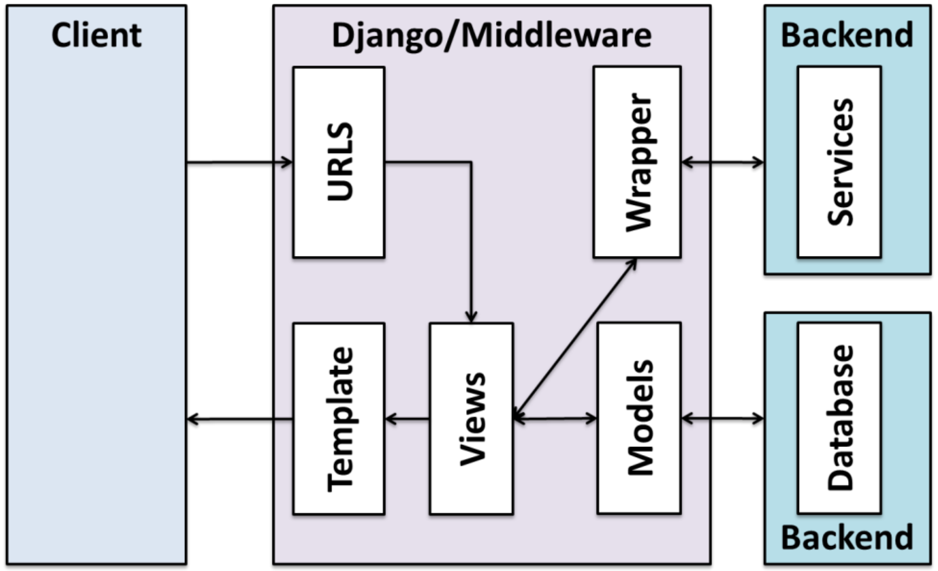
models.py

tests.py

urls.py

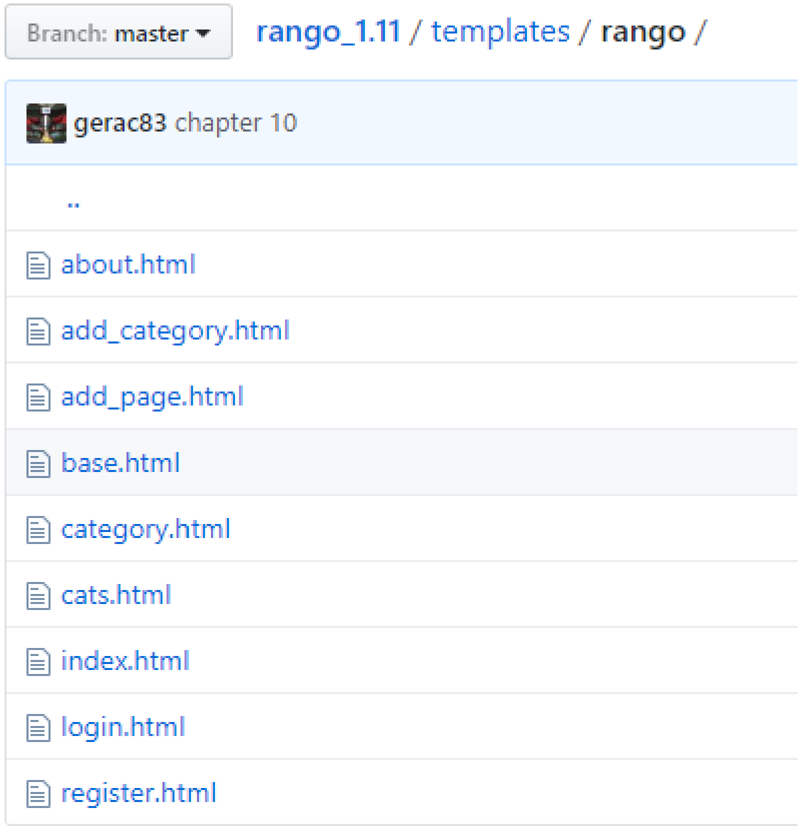
views.py

Internal Sections/Components



- **Providing Templates**
  - The templates mean the response format (html,xml,etc) is decoupled from the data to be presented

Internal Sections/Components



Branch: master

rango_1.11 / templates / rango /

gerac83 chapter 10

..

about.html

add_category.html

add_page.html

base.html

category.html

cats.html

index.html

login.html

register.html

Quiz Question (ID:1130)

Which of the following design philosophies does Django adhere to?

A. Tight coupling
B. Implicit is better than explicit
C. Don’t repeat yourself
D. All of the above

Go to: https://classresponse.gla.ac.uk/
Enter your GUID and password
Enter the session ID

Quiz Question (ID: 1130)

What command would you issue to download an existing GitHub repository (that you have not downloaded before)?

A. pull
B. push
C. clone
D. rebase

Go to: https://classresponse.gla.ac.uk/
Enter your GUID and password
Enter the session ID

QUICK INTRO TO HTML

What is HTML?

- HTML stands for **HyperText Markup Language**
- It’s the language web browsers use to interpret what gets displayed when you view a website
- Hypertext is the link between documents
- A mark-up language is a set of tags which describe document content
- HTML documents (webpages) contain HTML mark-up tags and plain text

Basic HTML Example

```
<html>          ← Begin HTML now
<head>
 <title> The title</title>
 </head>
<body>

<h1>ISD</h1>

 <p>My first paragraph.</p>

</body>
</html>         ← End HTML now
```

Tags:

- Keywords (tag names) surrounded by <>
- Normally have opening and closing tags

Plain text:

- Between tags
- Are the content displayed in the browser

Basic HTML Example

```
<!DOCTYPE html>
<html>
    <head>
      <title> The title</title>
    </head>
    <body>

      <h1>ISD</h1>

      <p>My first paragraph.</p>

    </body>
</html>
```

HTML Document Structure:

- Nested tags
- Starts with <html> tag
- <head> tag contains information about the document such as title and other things
- <body> tag contains the html to be displayed

Basic HTML Example

```
<!DOCTYPE html>
<html>
    <head>
      <title>The title</title>
    </head>
    <body>

      <h1>Things to do</h1>

      <p>My first paragraph.</p>

    </body>
</html>
```

Elements

- From an opening tag to a closing tag is called an element
- The plain text between opening and closing tags is called the element content

Basic HTML Example

```
<!DOCTYPE html>
<html>
    <head>
      <title>The title</title>
    </head>
    <body>

      <h1>Things to Do</h1>
       Make a HTML page<br>
       Add a paragraph
      <p>My first paragraph.</p>

    </body>
</html>
```

Empty Elements

- There are some tags which have no content
- They also have no end tag
- E.g. <br> which forces a line break

Basic HTML Example

```
<!DOCTYPE html>
<html>
    <head>
      <title> The title</title>
    </head>
    <body>

      <h1>Things to To</h1>

      <p>My first paragraph.</p>

    </body>
</html>
```

Two Basic Tags:

- <h1> - "header 1"
  - Used just once
  - Defines the most important heading
  - Search engines use H1 to determine the content of your web pages
  - There are h1,...,h6 headers. H1 being the most important.

Basic HTML Example

```
<!DOCTYPE html>
<html>
    <head>
      <title>The title</title>
    </head>
    <body>

      <h1>ISD</h1>

      <p>My first paragraph.</p>

    </body>
</html>
```

Two Basic Tags:

- <p> is the paragraph tag
- Browsers add space (margin) before and after each <p> element
- They ignore your own formatting

Other useful HTML tags

- List items / unordered list

```
<ul>
    <li> List item one</li>
    <li> List item two</li>
</ul>
```

- Div elements let you create sections to divide up the page in different ways when coupled with CSS.

```
<div> </div>
```

Quiz Question (ID: 1130)

Which of the following HTML tags is used at the start of a list item element?

A. <div>
B. <ul>
C. <li>
D. <li/>

Go to: https://classresponse.gla.ac.uk/
Enter your GUID and password
Enter the session ID