**Introduction to Servlets Quiz**

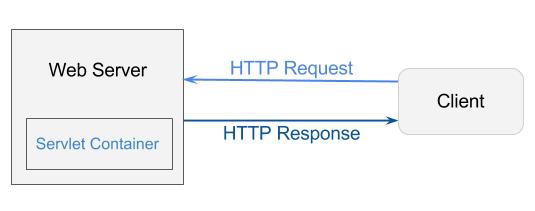
1. What is the difference between a web server and a web container?

**Web container** also known as a **Servlet container** is the component **of a web server** that interacts with Java servlets. ... **Web containers** are a part **of a web server** and they generally processes the user request and send a static response. **Servlet containers** are the **one** where JSP created components reside

1. What is a servlet?

The **Servlet** and the Container. Simply put, a **Servlet** is a class that handles requests, processes them and reply back with a response. For example, we can use a **Servlet** to collect input from a user through an HTML form, query records from a database, and create web pages dynamically

1. How do web servers and web containers interact with servlets?



[**Web container**](https://en.wikipedia.org/wiki/Web_container), specially in java, should be refer to servlet container. A servlet container is the component of a web server that interacts with java servlets. A web container is responsible for managing the life-cycle of servlets, mapping a URL to a particular servlet and ensuring that the URL requester has the correct access rights and many more such services. Basically, putting together all above facts, **servlet container is the runtime environment where your servlet run** and it’s life cycle is maintained

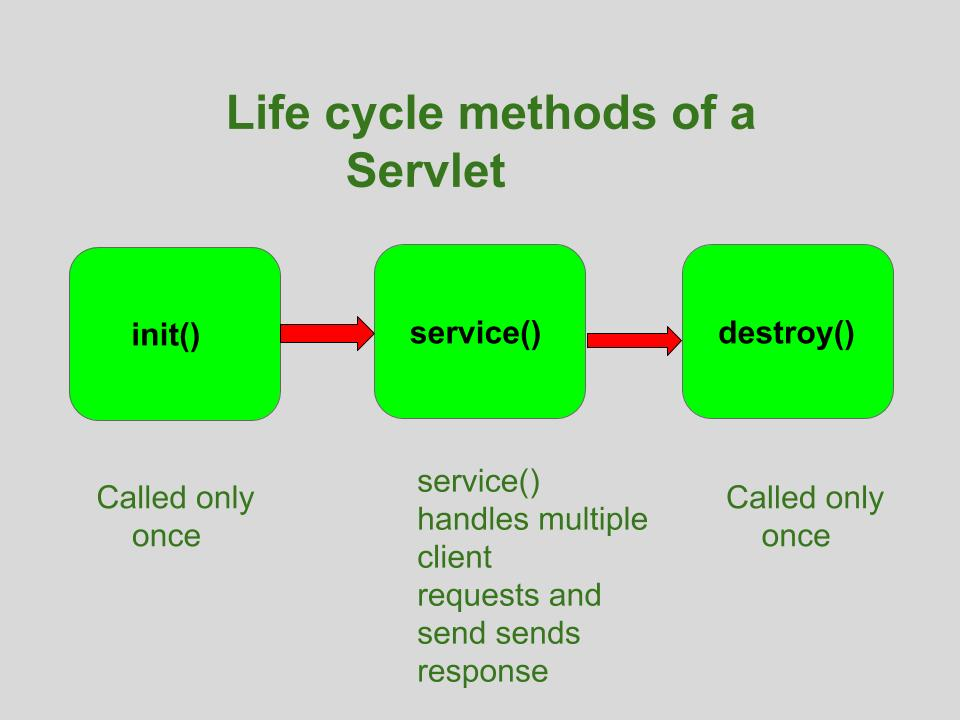
1. Who creates request objects?

When a client sends a request to the web server, the servlet container creates ServletRequest & ServletResponse objects and passes them as an argument to the servlet’s service() method. The request object provides the access to the request information such as header and body information of request data.

1. What are the states in the servlet lifecycle?

There are three life cycle methods of a Servlet :

* init()
* service()
* destroy()



1. Who calls init and when?

The **init** method is designed to be **called** only once. It is **called** when the servlet is first created, and not **called** again for each user request. So, it is used for one-time initializations, just as with the **init** method of applets

1. Which of init, service, and doGet should you override?

If **you** have **initialization** code (like getting a database connection or registering yourself with other objects), then **you**'ll **override** the **init**() method in your servlet class. ... The **doGet**() and doPost() inside HttpServlet don't **do** anything, so **you** have to **override one** or both

1. In what sense are servlets multi-threaded?

**Servlets** are intrinsically **multithreaded**.

This means a single instance can be accessed by more than one **thread**. If a **servlet** implements the SingleThreadModel interface, the container will not execute the service () method in more than one **thread** simultaneously

1. What are the implications of this for servlet instance variables?

The container does not create a new **instance** of the **Servlet** class for each request. It reuses an existing one. This is why they are not thread safe. The Stripes Action Framework DOES create a new **instance** for each request, so that's an okay assumption under that framework.