

Servlet basics

Nguyễn Hoàng Anh
Trương Phước Lộc
Hồ Tuấn Thanh

1

Khoa CNTT-ĐH.KHTN-2016

Agenda

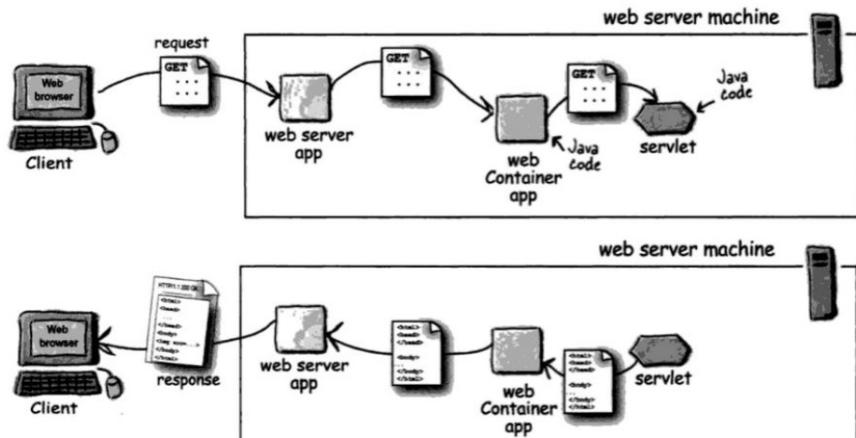
1. Container
2. Servlet
3. Story: build a site
4. Demo
5. Notes

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

2

2

1.1 What is a container?



Trương Phước Lộc – tploc@fit.hcmus.edu.vn

3

3

1.2 What does the Container give you?

- Communicate support
- Lifecycle management
- Multithreading support
- Declarative security
- JSP support

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

4

4

Khoa CNTT-ĐH.KHTN-2016

1.3 How the Container handles a request

The diagram shows a client (Web browser) sending an HTTP request (GET) to a container. The container then creates a servlet and two objects: HttpServletRequest and HttpServletResponse.

- ① Client sends HTTP request (GET) to container.
- ② Container sees it's for a servlet, so it creates two objects: HttpServletRequest and HttpServletResponse.

User clicks a link that has a URL to a servlet instead of a static page.

The container "sees" that the request is for a servlet, so the container creates two objects:
1) HttpServletRequest
2) HttpServletResponse

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

5

5

Khoa CNTT-ĐH.KHTN-2016

1.3 How the Container handles a request

The container finds the correct servlet based on the URL in the request, creates or allocates a thread for that request, and passes the request and response objects to the servlet thread.

The container calls the servlet's service() method. Depending on the type of request, the service() method calls either the doGet() or doPost() method.

For this example, we'll assume the request was an HTTP GET.

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

6

6

Khoa CNTT-ĐH.KHTN-2016

1.3 How the Container handles a request

The diagram illustrates the process of a container handling a request from a client web browser. It shows two sequential steps:

- Step 1:** A client (Web browser) sends a request to a container. The container delegates the request to a servlet. The servlet's `service()` method calls `doGet()`. The `doGet()` method generates a dynamic page and places it into a response object. A note states: "The doGet() method generates the dynamic page and stuffs the page into the response object. Remember, the container still has a reference to the response object!"
- Step 2:** The thread completes. The container converts the response object into an HTTP response, sends it back to the client, and then deletes the request and response objects. A note states: "The thread completes, the container converts the response object into an HTTP response, sends it back to the client, then deletes the request and response objects."

Truong Phuoc Loc – tploc@fit.hcmus.edu.vn

7

Khoa CNTT-ĐH.KHTN-2016

Agenda

- 1. Container
- 2. Servlet
- 3. Story: build a site
- 4. Demo
- 5. Notes

Truong Phuoc Loc – tploc@fit.hcmus.edu.vn

8

2. Servlet

- **Servlets** provide a component-based, platform-independent method for building Web-based applications.
- **Servlets** have access to the **entire family of Java APIs**, including the JDBC API to access enterprise databases

2.1 How it looks in code

In the real world, 99.9% of all servlets override either the `doGet()` or `doPost()` method.

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class Ch2Servlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
        throws IOException {
        PrintWriter out = response.getWriter();
        java.util.Date today = new java.util.Date();
        out.println("<html> " +
                   "<body> " +
                   "<h1 style='text-align:center>" +
                   "HFV's Chapter2 Servlet</h1>" +
                   "<br>" + today +
                   "</body>" +
                   "</html>");
    }
}
```

Notice... no `main()` method.
The servlet's lifecycle methods (like `doGet()`) are called by the Container.

99.9999% of all servlets are `HttpServlets`.

This is where your servlet gets references to the request and response objects which the container creates.

You can get a `PrintWriter` from the `response` object your servlet gets from the Container. Use the `PrintWriter` to write HTML text to the `response` object. (You can get other output options, besides `PrintWriter`, for writing, say, a picture instead of HTML text.)

Khoa CNTT-DH.KHTN-2016

2.2 A servlet can have THREE names

The diagram shows three stages of a developer's thought process:

- Client-known URL name:** A developer sits at a desk thinking, "I'll click the link to the 'register/registerMe' servlet."
- Deployer-known secret internal name:** The developer continues, "I'm gonna call this servlet the 'EnrollServlet'."
- Actual file name:** The developer finally realizes, "Wait a minute... I need to make sure this matches the actual file name: 'SignUpServlet.class'."

Truong Phuoc Loc – tploc@fit.hcmus.edu.vn 11

11

Khoa CNTT-DH.KHTN-2016

2.2 A servlet can have THREE names

- ① **<servlet>**
maps internal name to fully-qualified class name
- ② **<servlet-mapping>**
maps internal name to public URL name

This web app has two servlets.

There is a LOT more that goes into this opening <web-app> tag, but we don't want to show it right now (there's an example at the end of this chapter).

The <servlet> element tells the Container which class files belong to a particular web application.

```

<web-app ...>
  <servlet>
    <servlet-name>Internal name 1</servlet-name>
    <servlet-class>foo.Servlet1</servlet-class>
  </servlet>
  <servlet>
    <servlet-name>Internal name 2</servlet-name>
    <servlet-class>foo.Servlet2</servlet-class>
  </servlet>

```

The <servlet-name> element is used to tie a <servlet> element to a specific <servlet-mapping> element. The end-user NEVER sees this name; it's used only in other parts of the DD.

You put in the fully-qualified name of the class (but you don't add the ".class" extension).

Truong Phuoc Loc – tploc@fit.hcmus.edu.vn 12

12

2.2 A servlet can have THREE names

Think of the `<servlet-mapping>` element as what the Container uses at runtime when a request comes in, to ask, "which servlet should I invoke for this requested URL?".

```

<servlet-mapping>
  <servlet-name>Internal name 1</servlet-name>
  <url-pattern>/Public1</url-pattern>
</servlet-mapping>

<servlet-mapping>
  <servlet-name>Internal name 2</servlet-name>
  <url-pattern>/Public2</url-pattern>
</servlet-mapping>

</web-app>

```

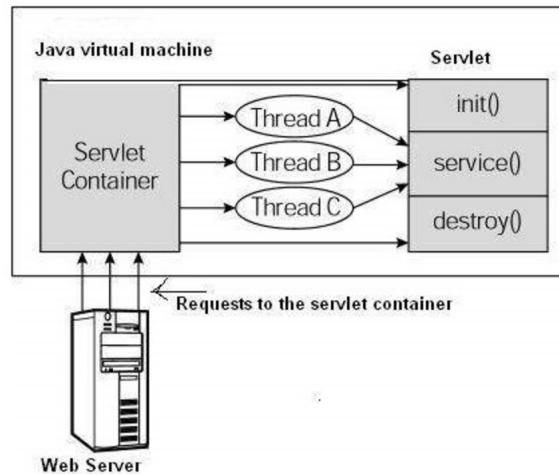
This is the what the client sees (and uses) to get to the servlet... but it's a made-up name that is NOT the name of the actual servlet class.

It's possible to use wildcards in the `<url-pattern>` element... more on that and paths later.

2.3 Parameters

- String str = request.getParameter("nameInput")
- String []s = request.getParameterValues("NameInput")
- Map<String, String[]> m = request.getParameterMap()
- Enumeration<String> e = request.getParameterNames()

2.4 Life Cycle



Trương Phước Lộc – tploc@fit.hcmus.edu.vn

15

15

2.4 Life Cycle

- The servlet is initialized by calling the **init()** method.
- The servlet calls **service()** method to process a client's request.
- The servlet is terminated by calling the **destroy()** method.
- Finally, servlet is garbage collected by the garbage collector of the JVM.

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

16

16

2.4 The init() method

- The init method is called **only once**.
- It is called only when the servlet is created, and not called for any user requests afterwards.

```
public void init() throws ServletException {
    //Initialization code...
}
```

2.4 The service() method

- The service() method is the **main method to perform the actual task**.
- The servlet container (i.e. web server) calls the service() method to
 - handle requests coming from the client(browsers)
 - write the formatted response back to the client.
- Each time the server **receives a request** for a servlet, the server spawns a new thread and calls service. The service() method **checks the HTTP request type** (GET, POST, PUT, DELETE, etc.) and calls doGet, doPost, doPut, doDelete, etc. methods as appropriate.

2.4 The destroy() method

- The `destroy()` method is called **only once** at the end of the life cycle of a servlet.
- This method gives your servlet a chance to close database connections, halt background threads, write cookie lists or hit counts to disk, and perform other such cleanup activities.

Agenda

1. Container
2. Servlet
3. Story: build a site
4. Demo
5. Notes

Khoa CNTT-ĐH.KHTN-2016

3. Bob builds a Matchmaking Site

I want an Agile
Dating site where
geeks can meet and hook up.
Because not everybody
gets lucky at a Linux
Installathon...

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

21

21

Khoa CNTT-ĐH.KHTN-2016

3. Bob builds a Matchmaking Site

GeekDates
"78% of our transactions end in commit."

Join
DQL query
Refactor my Profile

Input your state
Handle
Age
OS
Attributes
Exceptions
Type declaration
Insert!

Query
Compose your Dating Query Language (DQL) string here:
Do it

DQL Query Results
More

Refactor
Modify your profile:
[profile here]
Improve it

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

22

22

Khoa CNTT-ĐH.KHTN-2016

3. ... one for each page

```

// import statements

public class DatingServlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
        throws IOException {
        // business logic goes here, depending
        // on what this servlet is supposed to do
        // (write to the database, do the query, etc.)
        PrintWriter out = response.getWriter();
        // compose the dynamic HTML page
        out.println("something really ugly goes here");
    }
}

```

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

23

23

Khoa CNTT-ĐH.KHTN-2016

3.

The servlet does whatever it needs to do to process the request (like insert or search the database) and returns the HTML page in the HTTP response.

All of the business logic AND the client HTML page response is inside the servlet code.

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

24

24

Khoa CNTT-ĐH.KHTN-2016

Agenda

- 1. Container
- 2. Servlet
- 3. Story: build a site
- 4. Demo
- 5. Notes

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

25

25

Khoa CNTT-ĐH.KHTN-2016

4. Demo

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

26

26

Khoa CNTT-ĐH.KHTN-2016

Agenda

1. Container
2. Servlet
3. Story: build a site
4. Demo
5. Notes

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

27

27

Khoa CNTT-ĐH.KHTN-2016

5.1 The deployment descriptor (DD)



DD Benefits

- Minimizes touching source code that has already been tested.
- Lets you fine tune your app's capabilities, even if you don't have the source code.
- Lets you adapt your application to different resources (like databases), without having to recompile and test any code.
- Makes it easier for you to maintain dynamic security info like access control lists and security roles.
- Lets non-programmers modify and deploy your web applications while you can focus on the more interesting things. Like how appropriate your wardrobe isn't for a trip to Hawaii.

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

28

28

5.2 Servlet

- Form Data
- Client request
- Server response
- Filters
- Exceptions
- Cookies
- Session
- Database
- ...

Hỏi đáp



Khoa CNTT-ĐH.KHTN-2016

Tài liệu tham khảo

- Tài liệu giảng dạy – ThS. Nguyễn Hoàng Anh,
ĐH. Khoa học tự nhiên
- <http://www.tutorialspoint.com/servlets/>

Trương Phước Lộc – tploc@fit.hcmus.edu.vn

31

31