UNIT-3

Java Beans:

Introduction, JavaBeans Properties, Examples

Struts 2: Basic MVC Architecture

Struts 2 framework features

Struts 2 MVC pattern

Request life cycle Examples,

Configuration Files, Actions, Interceptors, Results & Result Types, Value Stack/OGNL

•**JSON:** Overview, Syntax, DataTypes, Objects, Schema, Comparison with XML, JSON with Java

What is Struts?

- is a framework to develop web application easily.
- makes easier to develop web application and maintain them.
- is an open source framework which makes building web applications easier, based on Java Servlets and JSP technologies.
- The Struts framework was created by Craig R. McClanahan and was donated to the Apache software foundation in 2000. Since then it is a open source software.

Why struts? What's wrong with jsp/servlet coding?

- **Using only Servlets** – difficult to output a html and needs lot of out.printlns – hard to read and clumsy
- **Using only JSP** – added scriptlets and implicit objects into jsp - awkward to see java inside html– hard to read and maintain – useful if very small application
- **Using JSP+ Java beans** – Code inside bean and jsp to display . Good choice for small applications. But what if there is need of multiple type of views? Eg: if there is need of different language display depending on client location? - making request to a general servlet, which outputs data according to the client locale, for same url request, will be good choice – Model 2 architecture evolved.
- **Using JSP+Servlets+JavaBeans** → Model 2 architecture

  Request made to servlet, servlet does business calculation using simple java POJO gets the result. Also decides the view and give back the response using the view to the client.

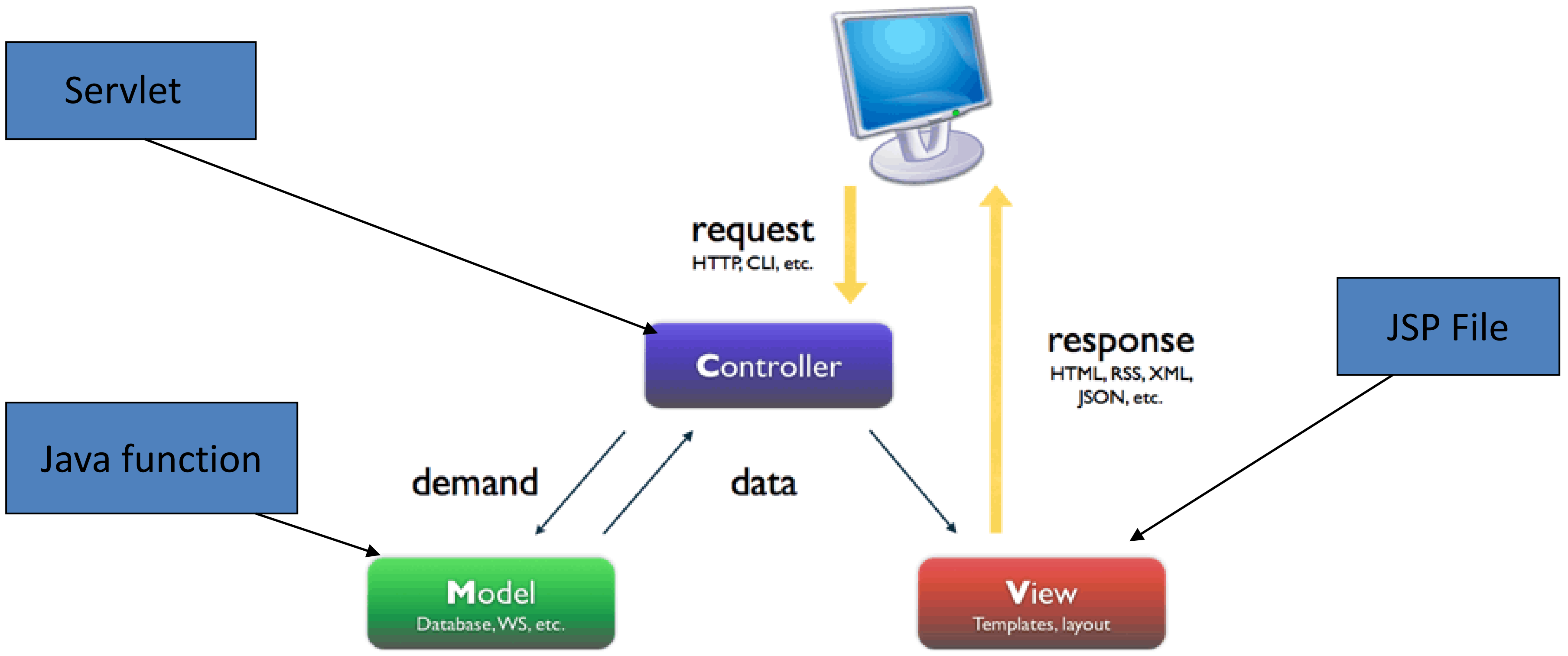
  Here servlet called – Controller, Business calculation POJO called – Model and JSP called - View

  Uses : the business logic is separated from JSPs and JSP gets displayed depending upon the result of model (the business function). → similar behavior like all applications above, but the code is more structured now. Changing business logic will not affect view and vice versa.

Struts 2.0 also uses Model 2 MVC pattern

Still the question remains: Why struts?

Stucture of JSP+Servlets+JavaBeans :Model 2 architecture



Why do we need Web Application Frameworks like Struts when we have good solution like Model 2 pattern?

Struts will be obviously complicated than Model 2 even though it uses same MVC.
Why do we make our application complicated using struts?

Before answering this lets check what is a framework

What is a Web framework?

- Web framework is a basic readymade underlying structure, where you have to just add components related to your business.
  - For example, if you take struts, it comes with all jars you might need to develop basic request response cycle, and with basic configuration. It provides the controller servlet or filter which will read your request and convert it to integer or float etc according to your business requirements. If there is any exception while converting, you don't have to deal with that. Framework deals with the problem and displays you exact message.
  - After all conversion, the framework automatically populate all your data needed from form to the java object.
  - You don't have to write much code to validate all your form data. Frame work provides some basic automatic validations.
  - After you write business logic, you don't have to write code to dispatch request to another page. Etc
- It forces the team to implement their code in a standard way. (helps debugging, fewer bugs etc).
  - For Example, in struts immediate backend logic should be in action classes's method. Action class functions intern can call other components to finish business logic.
- Framework might also help you develop complex User Interface easily like iterating tables, the menu etc (provide some tag libraries)
- Using frameworks like struts 2.0, one need not have to have deep knowledge of Http protocol and its request and responses interfaces to write business logic

Why Application Frameworks?

- Most Model 2 architecture based web applications share a common set of functionality. For example, they all do receive and dispatch HTTP requests, invoking model methods, selecting and assembling views.
- If everybody is doing the same thing over and over every time - Spending less time on business logic.
- Its good idea to have a common framework that support these set of functionalities
- only thing developer have to do is basically using or extending the frameworks using common interface and classes and can concentrate on business logic.
- Provides rich set of features

When to use what?

- If your application is very small and have only few pages, implementing Model 2 pattern by coding servlets / JSP / and java beans would be easier and simpler.
- If Application is large have got complicated flow and business logic, need to be maintained for long terms, using frameworks would save time and resource.
- Using frameworks like struts make its easy to maintain and develop and application will be more structured.

Why struts?

- Free to develop & deploy –open source
- Stable & Mature
- Many supported third-party tools
- Feature-rich
- Flexible & Extendable
- Large User Community, Expert Developers and Committers
- Rich tag library (html, bean tags etc)
- Easy to test and debug

How does struts make code simpler? A Sample Jsp / servlet code:

- your application might have to do following in you beans or in jsp to get a value of user input in double:

```
Jsp file: <input name="txtAmount"> </input>
In Bean or Jsp file:
String strAmount = request.getParameter("txtAmount");
double amount = 0.0;
try{
   double amount = Double.parseDouble(strAmount );
}catch(Exception e){
  // got error so return back.
// Big code for dispatching – also used in several places
  // return back to same page - hard coded
}

bean.setAmout(amount);
boolean flgResult = ejbObject.processAmount(amount);;
if(flgResult){
// success
// dispatch request to same or different page - hard coded
}else{
 // dispatch request to same or different page - hard coded
}
```

Using web framework like struts 2.0 it will look simpler

Jsp file:

```
<s:textfield label="Amount" name="amount" value="%{amount}" />
```

In action file you must have simple getter and setter:

```
double amount;
public double getAmount(){ return amount;}
public void setAmount(double amount){this.amount = amount;}
```

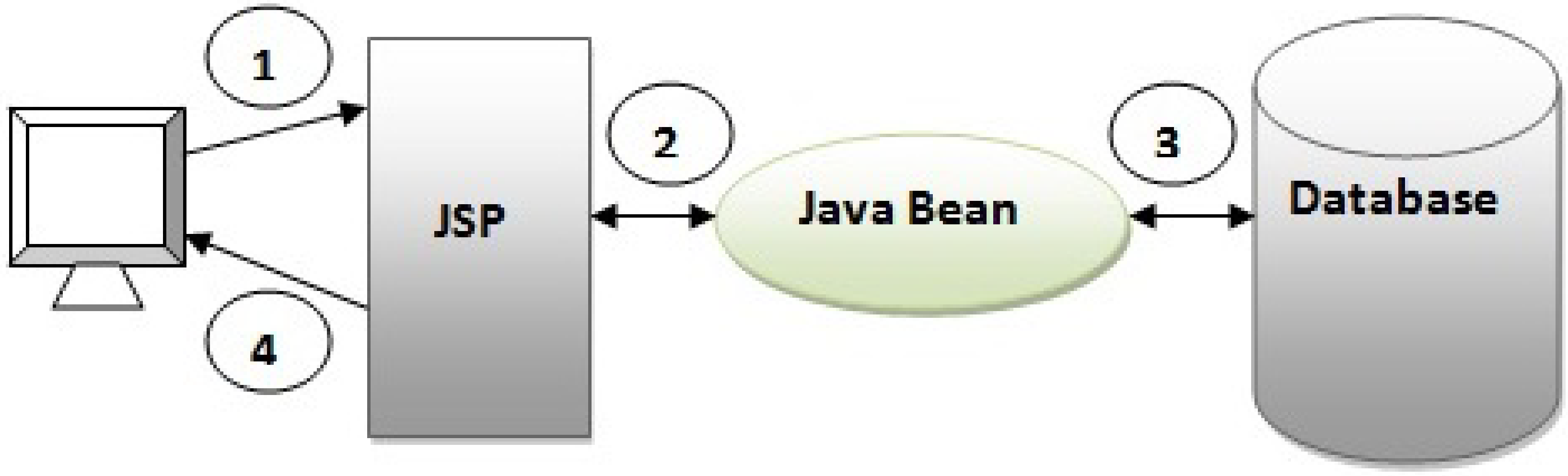
That's it. You can directly use the amount in action method without get extra code:

```
public String execute() throws Exception{
  // use amount directly
return "success";
}
```

Also there is no need of extra code for forwarding request.Action method is just returning a string "success"

Struts Framework Features

- Model 2 -MVC Implementation
- Internationalization(I18N) Support
- Rich JSP Tag Libraries
- Annotation and XML configuration options
- POJO-based actions that are easy to test
- Based on JSP, Servlet, XML, and Java
- Less xml configuration
- Easy to test and debug with new features
- Supports Java’s Write Once, Run Anywhere Philosophy
- Supports different model implementations (JavaBeans, EJB, etc.)
- Supports different presentation implementations(JSP, XML/XSLT, etc)



Model 1 Architecture

- *Browser sends request for the JSP page*
- *JSP accesses Java Bean and invokes business logic*
- *Java Bean connects to the database and get/save data*
- *Response is sent to the browser which is generated by JSP*

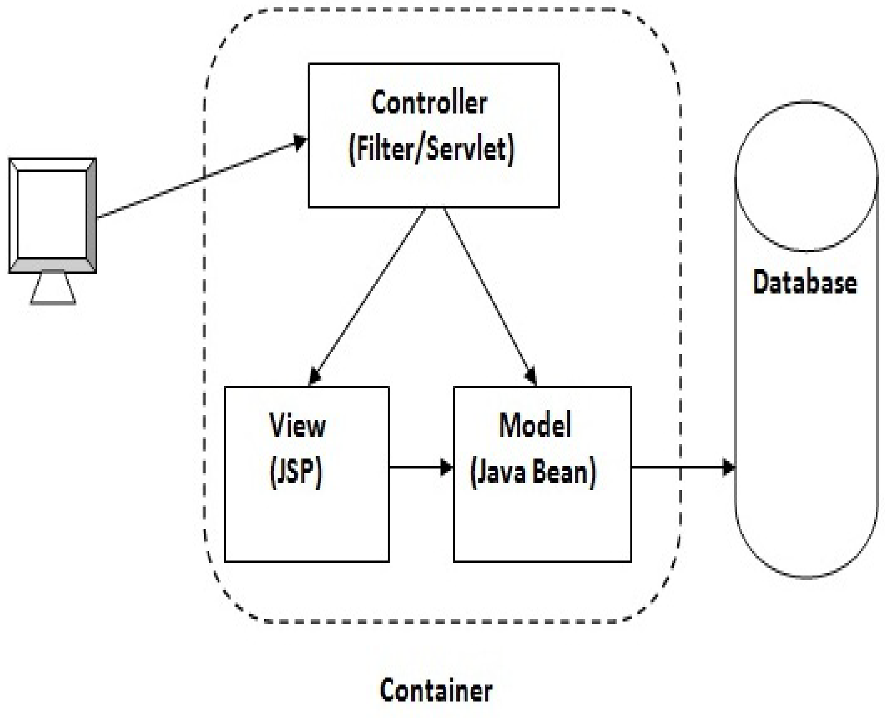
Advantage & Disadvantage of Model 1 Architecture

- Easy and Quick to develop web application

Disadvantage of Model 1 Architecture

- **Navigation control is decentralized** since every page contains the logic to determine the next page. If JSP page name is changed that is referred by other pages, we need to change it in all the pages that leads to the maintenance problem.
- **Time consuming** You need to spend more time to develop custom tags in JSP. So that we don't need to use scriptlet tag.
- **Hard to extend** It is better for small applications but not for large applications.

Model 2



Advantage of Model 2 (MVC)

- **Navigation control is centralized** Now only controller contains the logic to determine the next page.
- **Easy to maintain**
- **Easy to extend**
- **Easy to test**
- **Better separation of concerns**
- Disadvantage of Model 2 (MVC)
- We need to write the controller code self. If we change the controller code, we need to recompile the class and redeploy the application.

What are the pieces of struts?

- The filter dispatcher, by which all the requests to the applications gets filtered. - Controller
- The interceptors which called after filters, before action methods, apply common functionalities like validation, conversion etc
- Action classes with execute methods, usually storing and/or retrieving information from a database the view,i.e the jsp files
- Result will be figured out and the jsp file will be rendered.

Lets look at sample Hello World application of struts

steps

ValueStack

- A valueStack is simply a stack that contains application specific objects such as action objects and other model object.
- At the execution time, action is placed on the top of the stack.
- We can put objects in the valuestack, query it and delete it.

Methods of ValueStack interface

- **public String findString(String expr)** finds the string by evaluating the given expression.
- **public Object findValue(String expr)** finds the value by evaluating the specified expression.
- **public Object findValue(String expr, Class c)** finds the value by evaluating the specified expression.
- **public Object peek()** It returns the object located on the top of the stack.
- **public Object pop()** It returns the object located on the top of the stack and removes it.
- **public void push(Object o)** It puts the object on the top of the stack.
- **public void set(String key, Object value)** It sets the object on the stack with the given key. It can be get by calling the findValue(key) method.
- **public int size()** It returns the number of objects from the stack.

OGNL

- The **Object Graph Navigation Language** (OGNL) is an expression language.
- The struts framework sets the **ValueStack** as the root object of OGNL. Notice that action object is pushed into the ValueStack. We can direct access the action property.
- <s:property value="username"/>
- Here, username is the property key.
- The struts framework places other objects in ActionContext also e.g. map representing the **request**, **session**, **application** scopes.
- To get these values i.e. not the action property, we need to use # notation. For example to get the data from session scope, we need to use #session as given in the following example:
- <s:property name="#session.username"/>
- (or)
- <s:property name="#session['username']"/>

Struts 2 Action

- In struts 2, action class is **POJO** (Plain Old Java Object).
- POJO means you are not forced to implement any interface or extend any class.
- Generally, **execute** method should be specified that represents the business logic. The simple action class may look like:

Welcome.java

```
package com.javatpoint;
public class Welcome {
public String execute(){
   return "success";
}
}
```

- Action Interface
- A convenient approach is to implement the **com.opensymphony.xwork2.Action** interface that defines 5 constants and one execute method.
- 5 Constants of Action Interface
- Action interface provides 5 constants that can be returned form the action class. They are:
- **SUCCESS** indicates that action execution is successful and a success result should be shown to the user.
- **ERROR** indicates that action execution is failed and a error result should be shown to the user.
- **LOGIN** indicates that user is not logged-in and a login result should be shown to the user.
- **INPUT** indicates that validation is failed and a input result should be shown to the user again.
- **NONE** indicates that action execution is successful but no result should be shown to the user.

Actions

Welcome.java

```
package com.javatpoint;
import com.opensymphony.xwork2.Action;
public class Welcome implements Action{
public String execute(){
   return SUCCESS;
}
}
```

- **ActionSupport class**
- It is a convenient class that implements many interfaces such as Action, Validateable, ValidationAware, TextProvider, LocaleProvider and Serializable . So it is mostly used instead of Action.
-

Struts 2 Configuration File

- The struts application contains two main configuration files
- **struts.xml** file and **struts.properties** file.

- <?xml version="1.0" encoding="UTF-8" ?>
- <!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD Struts
- Configuration 2.1//EN" "http://struts.apache.org/dtds/struts-2.1.dtd">
- <struts>
- <**package** name="default" **extends**="struts-default">
-
- <action name="product" **class**="com.stud.Product">
- <result name="success">welcome.jsp</result>
- </action>
-
- </**package**>
- </struts>

Result Types in Struts2

- **<results>** tag plays the role of a **view** in the Struts2 MVC framework.
- The action is responsible for executing the business logic. The next step after executing the business logic is to display the view using the **<results>** tag.
- there are three possible outcomes Successful Login, Unsuccessful Login - Incorrect username or password

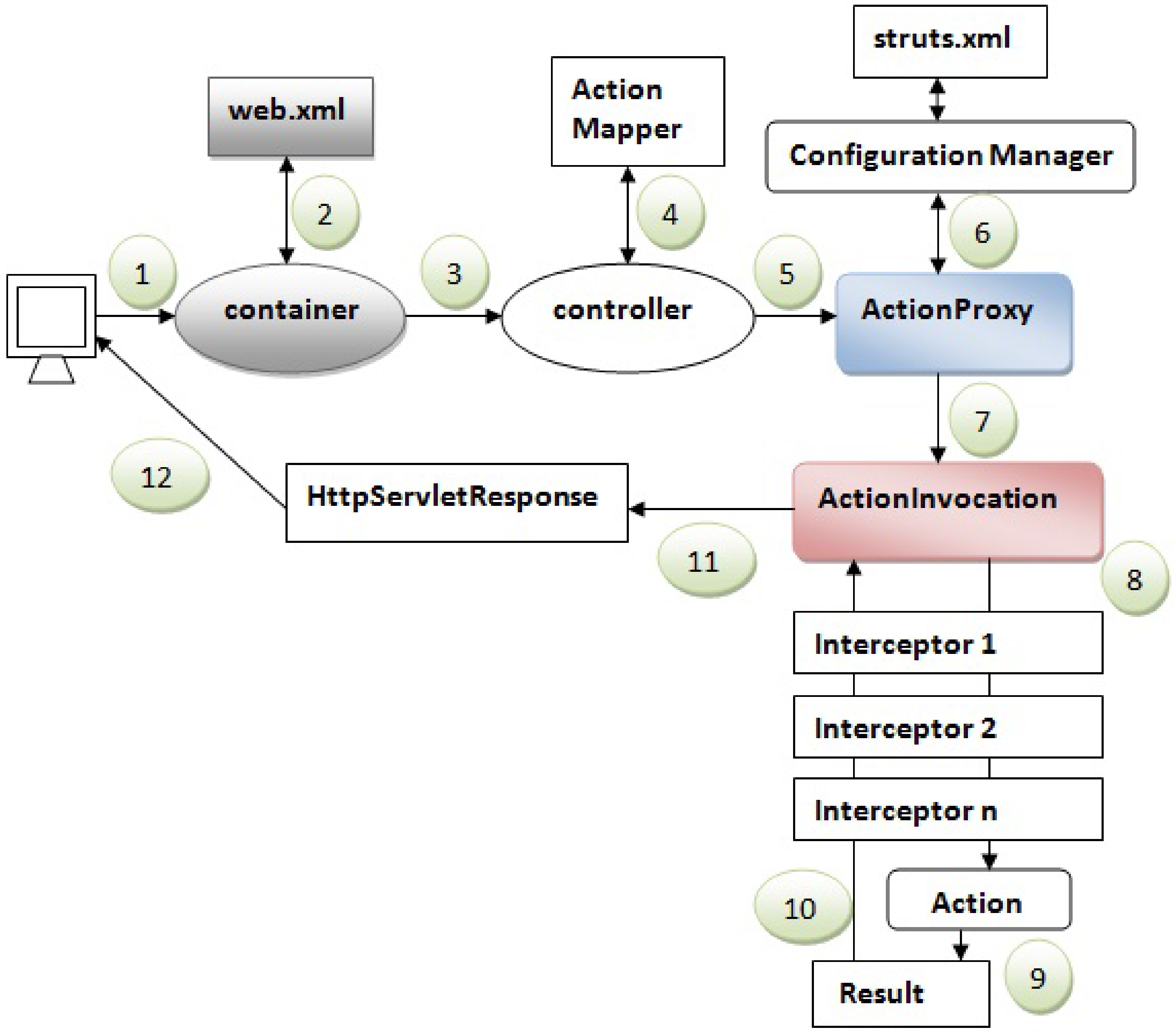
Result Types in Struts2

- Struts comes with a number of predefined **result types** and whatever we've already seen that was the default result type **dispatcher**, which is used to dispatch to JSP pages.
- Struts allow you to use other markup languages for the view technology to present the results and popular choices include **Velocity, Freemaker, XSLT** and **Tiles**.
- The **dispatcher** result type is the default type, and is used if no other result type is specified. It's used to forward to a servlet, JSP, HTML page, and so on, on the server. It uses the *RequestDispatcher.forward()* method.
- We saw the "shorthand" version in our earlier examples, where we provided a JSP path as the body of the result tag.

```
<result name = "success"> /HelloWorld.jsp </result>

<result name = "success" type = "dispatcher">
 <param name = "location"> /HelloWorld.jsp </param >
```

Struts2 architecture flow



struts.xml
web.xml
Action Mapper
Configuration Manager
1
2
3
4
5
6
container
controller
ActionProxy
7
12
HttpServletResponse
ActionInvocation
11
8
Interceptor 1
Interceptor 2
Interceptor n
Action
10
9
Result