请求参数和范围对象

采用基本类型接收请求参数

- 在Action类中定义与请求参数同名的属性，struts2便能自动接收请求参数并赋予给同名属性。

```
public class DepartmentAction {
	private Integer deptno;
	private String dname;

	public Integer getDeptno() {
		return deptno;
	}
	public void setDeptno(Integer deptno) {
		this.deptno = deptno;
	}
	public String getDname() {
		return dname;
	}
	public void setDname(String dname) {
		this.dname = dname;
	}
	public String execute() {
		System.out.println("deptno=" + deptno);
		System.out.println("dname=" + dname);
		return "department.jsp";
	}
}
```

采用复合类型接收请求参数

- 在Action中声明一个类类型的属性，Struts2利用反射动态创建对象封装请求数据

```
import java.sql.Date;
public class Employee {
    private Integer empno;
    private String ename;
    private Date hiredate;

    get和set方法……
}
```

采用复合类型接收请求参数（续）

EmployeeAction.java类

```
package action;
import pojo.Employee;

public class EmployeeAction {
	private Employee emp;

	public Employee getEmp() {
		return emp;
	}
	public void setEmp(Employee emp) {
		this.emp = emp;
	}

	public String add() {
		System.out.println("增加员工Action被执行...");
		return "employee.jsp";
	}
}
```

采用复合类型接收请求参数（续）

employees.jsp页面

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<html>
	<head>
		<title>员工信息管理</title>
	</head>
	<body>
		<form action="employee/list_add.action" method="post">
			员工编号：
			<input type="text" name="emp.empno" />
			<br>
			员工姓名：
			<input type="text" name="emp.ename" />
			<br>
			入职日期：
			<input type="text" name="emp.hiredate" />
			<br>
			<input type="submit" value="提交">
		</form>
		<hr>
		编号：${emp.empno },姓名：${emp.ename },入职日期：${emp.hiredate }
	</body>
</html>
```

自定义类型转换器

- 如果是java.sql.Date类型或者是java.util.Date类型对于yyyy-mm-dd格式的请求字符串是可以自动类型转换，但是如果是yyyyMMdd格式的日期字符串就无法自动转换，就需要自定义类型转换器
- 自定义类型转换器分为两种：
  - 局部类型转换器
    - 在Action类所在的包下放置ActionClassName-conversion.properties文件，ActionClassName是Action的类名，后面的-conversion.properties是固定写法。
    - 例如：HelloDateAction-conversion.properties
    - 在properties文件中的内容为：属性名称=类型转换器的全类名
    - 例如：hireDate=action.DateTypeConverter
  - 全局类型转换器
    - 在WEB-INF/classes下放置xwork-conversion.properties文件 。
    - 在properties文件中的内容为：待转换的类型=类型转换器的全类名
    - 例如：java.sql.Date=action.DateTypeConverter

自定义类型转换器代码

```
import java.text.SimpleDateFormat;
import java.util.Map;
import com.opensymphony.xwork2.conversion.impl.DefaultTypeConverter;

public class DateTypeConverter extends DefaultTypeConverter {
	public Object convertValue(Map<String, Object> context, Object value,
				Class toType) {
		SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMdd");

		try {
				String[] values = (String[]) value;//value参数表示请求参数数组

				if (toType == java.sql.Date.class) {
					java.util.Date uDate = sdf.parse(values[0]);
					return new java.sql.Date(uDate.getTime());
				}

				if (toType == java.util.Date.class) {
					return sdf.parse((String) values[0]);
				}
		} catch (Exception e) {
				e.printStackTrace();
		}

		return null;
	}
}
```

访问request/session/application等范围属性

Struts2可以使用和Servlet API无关的方式设置范围属性

示例

```
public String scope() throws Exception{
   ActionContext ctx = ActionContext.getContext();
   ctx.getApplication().put("app", "应用范围");//往ServletContext里放入app
   ctx.getSession().put("ses", "session范围");//往session里放入ses
   ctx.put("req", "request范围");//往request里放入req
   return "scope";
}
```

JSP:

```
 <body>
    ${applicationScope.app} <br>
    ${sessionScope.ses}<br>
    ${requestScope.req}<br>
 </body>
```

获取HttpServletRequest / HttpSession / ServletContext / HttpServletResponse对象

方法一，通过ServletActionContext.类直接获取：

```
public String rsa() throws Exception{
	HttpServletRequest request = ServletActionContext.getRequest();
	ServletContext servletContext = ServletActionContext.getServletContext();
	HttpSession session = request.getSession();
	HttpServletResponse response = ServletActionContext.getResponse();
	return "scope";
}
```

方法二，实现指定接口，由struts框架运行时注入：

```
public class BaseAction implements ServletRequestAware, ServletResponseAware, ServletContextAware{
	private HttpServletRequest request;
	private ServletContext servletContext;
	private HttpServletResponse response;
	public void setServletRequest(HttpServletRequest req) {
		this.request=req;
	}
	public void setServletResponse(HttpServletResponse res) {
		this.response=res;
	}
	public void setServletContext(ServletContext ser) {
		this.servletContext=ser;
	}
}
```