1. ServletConfig 接口

容器初始化一个 Servlet 时，为该 Servlet 创建一个唯一的 `ServletConfig` 对象，并将这个 `ServletConfig` 对象通过 init 方法传递并保存在这个 Servlet 对象中

作用

- 用于获取 Servlet 初始化参数和 ServletContext 对象

接口定义

- `public abstract interface javax.servlet.ServletConfig`

主要方法

- `getInitParameter(String param)`：根据给定的名称返回匹配的初始化参数
- `getInitParameterNames(String param)`：返回一个 Enumeration 对象，包含了所有初始化参数
- `getServletContext()`：返回 ServletContext 对象
- `getServletName()`：返回 Servlet 名字

2. ServletContext 接口

定义

`ServletContext` 是 Servlet 上下文，代表当前 Servlet 运行环境，是 Servlet 与 Servlet 容器之间直接通信的接口

获取 ServletContext 对象的方式

- 通过 `ServletConfig` 接口的 `getServletContext()` 方法
- 通过 `GenericServlet` 抽象类的 `getServletContext()` 方法获得

2.1. 获取应用初始化参数

- `getInitParameter(String name)`：通过 name 返回给定的数
- `getInitParameterNames()`：返回一个包含了初始化参数的 Enumeration 对象

2.2. 存取应用域属性

- `setAttribute(String name, Object obj)`
    - 存入一个 `key-value` 对到 `ServletContext` 中
- `getAttribute(String name)`
    - 根据参数给定的属性名，返回 `Object` 类型对象
- `getAttributeNames()`

    - 返回一个 Enumeration 对象，该对象包含了所有存放在 ServletContext 中的属性名
- `removeAttribute(String name)`
- 根据参数指定的属性名，从 `ServletContext` 对象中删除匹配的属性

2.3. 获取 Web 应用信息

- `getContextPath()`：返回当前 Web 应用的根路径
- `getServletContextName()`：返回 Web 应用的名字
- `getRequestDispacher(String path)`：返回用于请求转发的 `RequestDispatcher` 对象
- `getContext(String uripath)`
- 根据参数指定 URL 返回当前 Servlet 容器中其它 Web 应用的 `ServletContext` 对象

2.4. 获取容器信息

- `getServerInfo()`：返回名字、版本
- `getMajorVersion()`：返回Api主版本号
- `getMinorVersion()`：次版本号
- `log(String msg)`：记录一般的日志
- `log(String message, Throwable throw)`：记录异常的堆栈日志

2.5. 获取服务器文件资源

- `getResourceAsStream(String path)`
    - 返回一个读取参数指定的文件的输入流，参数路径必须以 / 开头
- `getResource()`
    - 返回由 path 指定的资源路径对应的一个 URL 对象，参数路径必须以 / 开头
- `getRealPath(String path)`
    - 根据参数指定的虚拟路径，返回文件系统中的一个真实的路径
- `getMimeType(String file)`
- 返回参数指定的文件的 MIME 类型

3. HttpServletRequest 接口

HTTP协议

请求/响应模型

请求报文

- 请求行
- 请求头
- 请求正文
- `Servlet` 对象由 `Servlet` 容器在用户每次请求 `Servlet` 时创建并传入 `Servlet` 的 `service()` 方法中

作用

- 封装 HTTP 请求信息

3.1. 获取请求行信息

- `request.getMethod()`：请求使用的 `HTTP` 方法
- `request.getQueryString()`：URL 后面的查询字符串
- `request.getServletPath()`：Servlet 映射的路径
- `request.getContextPath()`：获取请求资源属于的 Web 应用的路径

3.2. 获取请求头信息

- `Host`：等协议信息等
- `Enumeration getHeaderNames()`：获得所有请求头名称
- `request.getHeader(String name)`：返回 name 指定的请求头域的值

3.3. 获取请求正文

- `post`：请求参数和值组成的字符串
- `get`：请求参数在请求行中
- `enctype`：用于指定表单的编码方式，将表单中的数据传送给 Web 服务器
    - `application/x-www-form-urlencoded`
        - `String getParameter(String name)`
        - `String [] getParameterValues(String name)`
        - `Map getParameterMap()`
    - `multipart/form-data`
        - 只适用 `post` 请求
        - `ServletInputStream getInputStream()`：上传文件的二进制输入流
        - `BufferedReader getReadeer()`：上传文件字符缓冲输入流

3.4. 获取请求域属性

- `void setAttribute(String name, Object value)`
- `Object getAttribute(String name)`

3.5. 请求参数中文问题

- `request.setCharactereEncoding(charset)`：未设置时服务器默认用 `ISO-8859-1` 对参数进行解码

get 中文问题

- 服务器配置文件 `server.xml` 中设置 Connector 元素的 URIEncoding 属性来指定解码字符集

4. HttpServletResponse 接口

作用：封装 `HTTP` 响应消息

4.1. 响应状态码

- `1xx`：表示信息，服务器收到请求，需要请求者继续执行操作
- `2xx`：请求已成功被服务器接受、理解、并接受
- `3xx`：表示需要客户端采取进一步的操作才能完成请求
- `4xx`：表示客户端错误
- `5xx`：表示服务器处理请求的过程有错误或者异常状态发生
- 常见
    - `200`：请求成功
    - `302`：资源暂时转移到其他 `url`
    - `404`：请求资源不存在
    - `500`：表示服务器内部错误

4.2. 设置响应状态的方法

- `setStatus(int sc)`：以指定的状态码将响应返回给客户端
- `setError(int sc)`：使用指定的状态码向客户端返回一个错误响应
- `sendError(int sc, String msg)`：使用指定状态码和状态描述向客户端返回一个错误响应
- `sendRedirect(String location)`：请求的重定向，会设定响应 `Location` 报头及改变状态码

4.3. 构建响应头的方法

- `setHeader(String location, String name)`
    - `location`：浏览器重新连接到的 `URL`，相当于使用 `sendRedirect()` 方法
- `addHeader(String location, String name)`：增加名为 name，值为 value 到 http 头部
    - `Cookie` 只能通过 add 添加，不能直接设置

4.4. 创建响应正文的方法

- `response.getOutputStream()`：返回字节流输出对象，主要是二进制字节数据
- `PrintWriter out = response.getWriter()`：返回字符输出流对象

响应中文问题

- `response.setCharacterEncoding("UTF-8")`

- `response.setContentType("text/html;charset=UTF-8")`