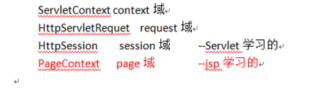


Serlvet四大域对象



1. page域:    只能在当前jsp页面使用                (当前页面)
2. request域: 只能在同一个请求中使用               (转发)
3. session域: 只能在同一个会话(session对象)中使用  (私有的)
4. context域: 只能在同一个web应用中使用            (全局的)

Pagecontxt对象可以获取其他八个内置对象





等价，pagecontext可以保存数据到其他域，也可以取值见下



## ****九大内置对象介绍****

****① out - javax.servlet.jsp.jspWriter****  
   out对象用于把结果输出到网页上。

方法：  
1. void clear() ;  
   清除输出缓冲区的内容，但是不输出到客户端。

2. void clearBuffer() ;  
   清除输出缓冲区的内容，并输出到客户端。

3. void close() ;  
   关闭输出流，清除所有内容。

4. void flush() ;  
   输出缓冲区里面的数据。

5. int getBufferSize() ;  
   获取以kb为单位的目前缓冲区大小。

6. int getRemaining() ;  
   获取以kb为单位的缓冲区中未被占用的空间大小。

7. boolean isAutoFlush() ;  
   是否自动刷新缓冲区。

8. void newLine() ;  
   输出一个换行字符。

9. void print( boolean b ) ;  
   void print( char c ) ;  
   void print( char[] s ) ;  
   void print( double d ) ;  
   void print( float f ) ;  
   void print( int i ) ;  
   void print( long l ) ;  
   void print( Object obj ) ;  
   void print( String s ) ;  
   将指定类型的数据输出到Http流，不换行。

10. void println( boolean b ) ;  
    void println( char c ) ;  
    void println( char[] s ) ;  
    void println( double d ) ;  
    void println( float f ) ;  
    void println( int i ) ;  
    void println( long l ) ;  
    void println( Object obj ) ;  
    void println( String s ) ;  
    将指定类型的数据输出到Http流，并输出一个换行符。  
      
11. Appendable append( char c ) ;  
    Appendable append( CharSequence cxq, int start, int end ) ;  
    Appendable append( CharSequence cxq ) ;  
    将一个字符或者实现了CharSequence接口的对象添加到输出流的后面。

成员：  
int DEFAULT\_BUFFER = 0    - 缺省缓冲区大小  
int NO\_BUFFER = -1        - writer是否处于缓冲输出状态  
int UNBOUNDED\_BUFFER = -2 - 是否限制缓冲区大小

****② request - javax.servlet.http.HttpServletRequest****   request对象包含所有请求的信息，如请求的来源、标头、cookies和请求相关的参数值等。

方法：  
1. Object getAttribute( String name ) ;  
   返回由name指定的属性值，该属性不存在时返回null。

2. Enumeration getAttributeNames() ;  
   返回request对象的所有属性名称的集合。

3. String getAuthType() ;  
   返回用来保护servlet的认证方法的名称，未受保护时返回null。

4. String getCharacterEncoding() ;  
   返回请求中的字符编码方法，可以在response对象中设置。

5. int getContentLength() ;  
   返回请求的BODY的长度，不能确定长度时返回-1。可以在response中设置。

6. String getContentType() ;  
   返回在response中定义的内容类型。

7. String getContentPath() ;  
   返回请求的路径。

8. Cookie[] getCookies() ;  
   返回客户端所有的Cookie的数组。

9. Enumeration getHeaderNames() ;  
   返回所有HTTP头的名称的集合。

10. Enumeration getHeaders( String name ) ;  
    返回指定HTTP头的所有值的集合。

11. String getHeader( String name ) ;  
    返回指定名称的HTTP头的信息。

12. long getDateHeader( String name ) ;  
    返回指定名称的Data类型的HTTP头的信息。

13. int getIntHeader( String name ) ;  
    返回指定名称的Int类型的HTTP头的信息。

14. ServletInputStream getInputStream() ;  
    返回请求的输入流。

15. Locale getLocale() ;  
    返回当前页的Locale对象，可以在response中设定。

16. Enumeration getLocales() ;  
    返回请求中所有的Locale对象的集合。

17. String getLocalName() ;  
    获取响应请求的服务器端主机名。

18. String getLocalAddr() ;  
    获取响应请求的服务器端地址。

19. int getLocalPort() ;  
    获取响应请求的服务器端端口

20. String getMethod() ;  
    获取客户端向服务器端发送请求的方法(GET、POST)。

21. String getParameter( String name ) ;  
    获取客户端发送给服务器端的参数值。

22. Map getParameterMap() ;  
    该方法返回包含请求中所有参数的一个Map对象。

23. Enumeration getParameterNames() ;  
    返回请求中所有参数的集合。

24. String[] getParameterValues( String name ) ;  
    获得请求中指定参数的所有值。

25. String getQueryString() ;  
    返回get方法传递的参数字符串，该方法不分解出单独的参数。

26. String getPathInfo() ;  
    取出请求中处于ServletPath和QueryString之间的额外信息。

27. String getPathTranslated() ;  
    返回用getPathInfo()方法取得的路径信息的实际路径。

28. String getProtocol() ;  
    返回请求使用的协议。可以是HTTP1.1或者HTTP1.0。

29. BufferedReader getReader() ;  
    返回请求的输入流对应的Reader对象，该方法和getInputStream()方法在一个页面中只能调用一个。

30. String getRemoteAddr() ;  
    获取发出请求的客户端IP地址。

31. String getRemoteHost() ;  
    获取发出请求的客户端主机名

32. String getRemoteUser() ;  
    返回经过客户端验证的用户名，未经验证返回null。

33. int getRemotePort() ;  
    返回发出请求的客户端主机端口。

34. String getRealPath( String path ) ;  
    返回给定虚拟路径的物理路径。

35. RequestDispatcher getRequestDispatcher( String path ) ;  
    按给定的路径生成资源转向处理适配器对象。

36. String getRequestedSessionId() ;  
    返回请求的session的标识。

37. String RequestURI() ;  
    返回发出请求的客户端地址，但是不包括请求的参数字符串。

38. StringBuffer getRequestURI() ;  
    返回响应请求的服务器端地址

39. String getScheme() ;  
    获取协议名称，缺省值为HTTP协议。

40. String getServerName() ;  
    返回响应请求的服务器名称。

41. String getServletPath() ;  
    获取客户端所请求的脚本文件的文件路径。

42. int getServerPort() ;  
    获取响应请求的服务器端主机端口号。

43. void removeAttribute( String name ) ;  
    在属性列表中删除指定名称的属性。

44. void setAttribute( String name, Object value ) ;  
    在属性列表中添加/删除指定的属性。

45. void setCharacterEncoding( String name ) ;  
    设置请求的字符编码格式。

46. HttpSession getSession() ;  
    HttpSession getSession( boolean create ) ;  
    获取session，如果create为true，在无session的情况下创建一个。  
      
47. boolean isRequestedSessionIdFromCookie() ;  
    检查请求的会话ID是否为通过Cookie传入。

48. boolean isRequestedSessionIdFromURL() ;  
    检查请求的会话ID是否为通过URL传入。

49. boolean isRequestedSessionIdValid() ;  
    检查请求的会话ID是否仍然有效。

50. boolean isSecure() ;  
    检查请求是否使用安全链接，如果HTTPS等。

51. boolean isUserInRole( String role ) ;  
    检查已经通过验证的用户是否在是role所指定的角色。

52. Principal getUserPrincipal() ;  
    返回包含用户登陆名的一个java.security.Principal对象。

成员：  
String BASIC\_AUTH = "BASIC"             -   
String CLIENT\_CERT\_AUTH = "CLIENT\_CERT" -   
String DIGEST\_AUTH = "DIGEST"           -   
String FORM\_AUTH = "FORM"               -

****③ response - javax.servlet.http.HttpServletResponse****  
   response对象主要将JSP容器处理后的结果传回到客户端。

方法：  
1. void addCookie( Cookie cookie ) ;  
   添加一个Cookie对象，保存客户端信息。

2. void addDateHeader( String name, long value ) ;  
   添加一个日期类型的HTTP头信息，覆盖同名的HTTP头信息。

3. void addHeader( String name, String value ) ;  
   添加一个HTTP头，覆盖同名的旧HTTP头。

4. void addIntHeader( String name, int value ) ;  
   添加一个整型的HTTP头，覆盖同名的旧HTTP头。

5. boolean containsHeader( String name ) ;  
   判断指定的HTTP头是否存在。

6. String encodeRedirectURL( String url ) ;  
   对sendRedirect()方法使用的URL进行编码。

7. String encodeURL( String url ) ;  
   将URL予以编码，回传包含session ID的URL。  
     
8. void flushBuffer() ;  
   强制把当前缓冲区的内容发送到客户端。

9. int getBufferSize() ;  
   取得以kb为单位的缓冲区大小。

10. String getCharacterEncoding() ;  
    获取响应的字符编码格式。

11. String getContentType() ;  
    获取响应的类型。

12. Locale getLocale() ;  
    获取响应的Locale对象。

13. ServletOutputStream getOutputStream() ;  
    返回客户端的输出流对象。

14. PrintWriter getWriter() ;  
    获取输出流对应的writer对象。

15. boolean isCommitted() ;  
    判断服务器端是否已经将数据输出到客户端。

16. void reset() ;  
    清空buffer中的所有内容。

17. void resetBuffer() ;  
    情况buffer中所有的内容，但是保留HTTP头和状态信息。

18. void sendError( int xc, String msg ) ;  
    void sendError( int xc ) ;  
    发送错误，包括状态码和错误信息。

19. void sendRedirect( String locationg ) ;  
    把响应发送到另外一个位置进行处理。

20. void setBufferSize( int size ) ;  
    设置以kb为单位的缓冲区大小。

21. void setCharacterEncoding( String charset ) ;  
    设置响应使用的字符编码格式。

22. void setContentLength( int length ) ;  
    设置响应的BODY长度。

23. void setContentType( String type ) ;  
    设置响应的类型。

24. void setDateHeader( String name, long value ) ;  
    设置指定名称的Data类型的HTTP头的值。

25. void setHeader( String name, String value ) ;  
    设置指定名称的HTTP头的值。

26. void setIntHeader( String name, int value ) ;  
    设置指定名称的int类型的HTTP头的值。

27. void setStatus( int xc ) ;  
    设置响应状态码，新值会覆盖当前值。

成员(HTTP状态码)：  
int SC\_CONTINUE = 100                      int SC\_SWITCHING\_PROTOCOLS = 101  
int SC\_OK = 200                            int SC\_NON\_AUTHORITATIVE\_INFORMATION = 203  
int SC\_ACCEPTED = 202                      int SC\_CREATED = 201  
int SC\_NO\_CONTENT = 204                    int SC\_RESET\_CONTENT = 205  
int SC\_PARTIAL\_CONTENT = 206               int SC\_MULTIPLE\_CHOICES = 300  
int SC\_MOVED\_PERMANENTLY = 301             int SC\_MOVED\_TEMPORARILY = 302  
int SC\_FOUND = 302                         int SC\_SEE\_OTHER = 303  
int SC\_NOT\_MODIFIED = 304                  int SC\_USE\_PROXY = 305  
int SC\_TEMPORARY\_REDIRECT = 307            int SC\_BAD\_REQUEST = 400  
int SC\_UNAUTHORIZED = 401                  int SC\_PAYMENT\_REQUIRED = 402  
int SC\_FORBIDDEN = 403                     int SC\_NOT\_FOUND = 404  
int SC\_METHOD\_NOT\_ALLOWED = 405            int SC\_NOT\_ACCEPTABLE = 406  
int SC\_PROXY\_AUTHENTICATION\_REQUIRED = 407 int SC\_REQUEST\_TIMEOUT = 408  
int SC\_CONFLICT = 409                      int SC\_GONE = 410  
int SC\_LENGTH\_REQUIRED = 411               int SC\_PRECONDITION\_FAILED = 412  
int SC\_REQUEST\_ENTITY\_TOO\_LARGE = 413      int SC\_REQUEST\_URI\_TOO\_LONG = 414  
int SC\_UNSUPPORTED\_MEDIA\_TYPE = 415        int SC\_REQUESTED\_RANGE\_NOT\_SATISFIABLE = 416  
int SC\_EXPECTATION\_FAILED = 417            int SC\_INTERNAL\_SERVER\_ERROR = 500  
int SC\_NOT\_IMPLEMENTED = 501               int SC\_BAD\_GATEWAY = 502  
int SC\_SERVICE\_UNAVAILABLE = 503           int SC\_GATEWAY\_TIMEOUT = 504  
int SC\_HTTP\_VERSION\_NOT\_SUPPORTED = 505

****④ session - javax.servlet.http.HttpSession****  
   session对象表示目前个别用户的会话状态，用来识别每个用户。

方法：  
1. Object getAttribute( String name ) ;  
   获取与指定名字相关联的session属性值。

2. Enumeration getAttributeNames() ;  
   取得session内所有属性的集合。

3. long getCreationTime() ;  
   返回session的创建时间，最小单位千分之一秒。

4. String getId() ;  
   取得session标识。

5. long getLastAccessedTime() ;  
   返回与当前session相关的客户端最后一次访问的时间，由1970-01-01算起，单位毫秒。

6. int getMaxInactiveInterval( int interval ) ;  
   返回总时间，以秒为单位，表示session的有效时间(session不活动时间)。-1为永不过期。

7. ServletContext getServletContext() ;  
   返回一个该JSP页面对应的ServletContext对象实例。

8. HttpSessionContext getSessionContext() ;

9. Object getValue( String name ) ;  
   取得指定名称的session变量值，不推荐使用。

10. String[] getValueNames() ;  
    取得所有session变量的名称的集合，不推荐使用。

11. void invalidate() ;  
    销毁这个session对象。

12. boolean isNew() ;  
    判断一个session是否由服务器产生，但是客户端并没有使用。

13. void pubValue( String name, Object value ) ;  
    添加一个session变量，不推荐使用。

14. void removeValue( String name ) ;  
    移除一个session变量的值，不推荐使用。

15. void setAttribute( String name, String value ) ;  
    设置指定名称的session属性值。

16. void setMaxInactiveInterval( int interval ) ;  
    设置session的有效期。

17. void removeAttribute( String name ) ;  
    移除指定名称的session属性。

****⑤ pageContext - javax.servlet.jsp.PageContext****  
   pageContext对象存储本JSP页面相关信息，如属性、内建对象等。

方法：  
1. void setAttribute( String name, Object value, int scope ) ;  
   void setAttribute( String name, Object value ) ;  
   在指定的共享范围内设置属性。

2. Object getAttribute( String name, int scope ) ;  
   Object getAttribute( String name ) ;  
   取得指定共享范围内以name为名字的属性值。

3. Object findAttribute( String name ) ;  
   按页面、请求、会话和应用程序共享范围搜索已命名的属性。

4. void removeAttribute( String name, int scope ) ;  
   void removeAttribute( String name ) ;  
   移除指定名称和共享范围的属性。

5. void forward( String url ) ;  
   将页面导航到指定的URL。

6. Enumeration getAttributeNamesScope( int scope ) ;  
   取得指定共享范围内的所有属性名称的集合。

7. int getAttributeScope( String name ) ;  
   取得指定属性的共享范围。

8. ErrorData getErrorDate() ;  
   取得页面的errorData对象。

9. Exception getException() ;  
   取得页面的exception对象。

10. ExpressionEvaluator getExpressionEvaluator() ;  
    取得页面的expressionEvaluator对象。

11. JspWriter getOut() ;  
    取得页面的out对象。

12. Object getPage() ;  
    取得页面的page对象。

13. ServletRequest getRequest() ;  
    取得页面的request对象。

14. ServletResponse getResponse() ;  
    取得页面的response对象。

15. ServletConfig getConfig() ;  
    取得页面的config对象。

16. ServletContext getServletContext() ;  
    取得页面的servletContext对象。

17. HttpSession getSession() ;  
    取得页面的session对象。

18. VariableResolver getVariableResolver() ;  
    取得页面的variableResolver对象。

19. void include( String url, boolean flush ) ;  
    void include( String url ) ;  
    包含其他的资源，并指定是否自动刷新。

20. void release() ;  
    重置pageContext内部状态，释放所有内部引用。

21. void initialize( Servlet servlet, ServletRequest request, ServletResponse response,  
                     String errorPageURL, boolean needSession, int bufferSize, boolean autoFlush ) ;  
    初始化未经初始化的pageContext对象。

22. BodyContext pushBody() ;  
    BodyContext pushBody( Writer writer ) ;  
    保存当前的out对象，并更新pageContext中page范围内的out对象。

23. JspWrite popBody() ;  
    取出由pushBody()方法保存的out对象。

24. void handlePageException( Exception e ) ;  
    void handlePageException( Thrwoable t ) ;

成员：  
int PAGE\_SCOPE = 1        - 页面共享范围  
int REQUEST\_SCOPE = 2     - 请求共享范围  
int SESSION\_SCOPE = 3     - 会话共享范围  
int APPLICATION\_SCOPE = 4 - 应用程序共享范围  
String PAGE = "javax.servlet.jsp.jspPage"  
String PAGECONTEXT = "javax.servlet.jsp.jspPageContext"  
String REQUEST = "javax.servlet.jsp.jspRequest"  
String RESPONSE = "javax.servlet.jsp.jspResponse"  
String CONFIG = "javax.servlet.jsp.jspConfig"  
String SESSION = "javax.servlet.jsp.jspSession"  
String OUT = "javax.servlet.jsp.jspOut"  
String APPLICATION = "javax.servlet.jsp.jspApplication"  
String EXCEPTION = "javax.servlet.jsp.jspException"

****⑥ application - javax.servlet.ServletContext****  
   application主要功用在于取得或更改Servlet的设定。

方法：  
1. Object getAttribute( String name ) ;  
   返回由name指定的application属性。

2. Enumeration getAttributes() ;  
   返回所有的application属性。

3. ServletContext getContext( String uripath ) ;  
   取得当前应用的ServletContext对象。

4. String getInitParameter( String name ) ;  
   返回由name指定的application属性的初始值。

5. Enumeration getInitParameters() ;  
   返回所有的application属性的初始值的集合。

6. int getMajorVersion() ;  
   返回servlet容器支持的Servlet API的版本号。

7. String getMimeType( String file ) ;  
   返回指定文件的类型，未知类型返回null。一般为"text/html"和"image/gif"。

8. int getMinorVersion() ;  
   返回servlet容器支持的Servlet API的副版本号。

9. String getRealPath( String path ) ;  
   返回给定虚拟路径所对应物理路径。

10. RequestDispatcher getNamedDispatcher( String name ) ;  
    为指定名字的Servlet对象返回一个RequestDispatcher对象的实例。

11. RequestDispatcher getRequestDispatcher( String path ) ;  
    返回一个RequestDispatcher对象的实例。

12. URL getResource( String path ) ;  
    返回指定的资源路径对应的一个URL对象实例，参数要以"/"开头。

13. InputStream getResourceAsStream( String path ) ;  
    返回一个由path指定位置的资源的InputStream对象实例。

14. Set getResourcePaths( String path ) ;  
    返回存储在web-app中所有资源路径的集合。

15. String getServerInfo() ;  
    取得应用服务器版本信息。

16. Servlet getServlet( String name ) ;  
    在ServletContext中检索指定名称的servlet。

17. Enumeration getServlets() ;  
    返回ServletContext中所有servlet的集合。

18. String getServletContextName() ;  
    返回本web应用的名称。

19. Enumeration getServletContextNames() ;  
    返回ServletContext中所有servlet的名称集合。

20. void log( Exception ex, String msg ) ;  
    void log( String msg, Throwable t ) ;  
    void log( String msg ) ;  
    把指定的信息写入servlet log文件。

21. void removeAttribute( String name ) ;  
    移除指定名称的application属性。

22. void setAttribute( String name, Object value ) ;  
    设定指定的application属性的值。

****⑦ config - javax.servlet.ServletConfig****   config对象用来存放Servlet初始的数据结构。

方法：  
1. String getInitParameter( String name ) ;  
   返回名称为name的促使参数的值。

2. Enumeration getInitParameters() ;  
   返回这个JSP所有的促使参数的名称集合。

3. ServletContext getContext() ;  
   返回执行者的servlet上下文。

4. String getServletName() ;  
   返回servlet的名称。

****⑧ exception - java.lang.Throwable****  
   错误对象，只有在JSP页面的page指令中指定isErrorPage="true"后，才可以在本页面使用exception对象。

方法：  
1. Throwable fillInStackTrace() ;  
   将当前stack信息记录到exception对象中。

2. String getLocalizedMessage() ;  
   取得本地语系的错误提示信息。

3. String getMessage()  
   取得错误提示信息。

4. StackTrackElement[] getStackTrace() ;  
   返回对象中记录的call stack track信息。

5. Throwable initCause( Throwable cause ) ;  
   将另外一个异常对象嵌套进当前异常对象中。  
     
6. Throwable getCause() ;  
   取出嵌套在当前异常对象中的异常。

7. void printStackTrace() ;  
   void printStackTrace( printStream s ) ;  
   void printStackTrace( printWriter s ) ;  
   打印出Throwable及其call stack trace信息。

8. void setStackTrace( StackTraceElement[] stackTrace )  
   设置对象的call stack trace信息。

****⑨ page - javax.servlet.jsp.HttpJspPage****  
   page对象代表JSP对象本身，或者说代表编译后的servlet对象，  
   可以用( (javax.servlet.jsp.HttpJspPage)page )来取用它的方法和属性。