1. How can a servlet access it’s associated ServletConfig object? (1 correct answer)
   1. getServletConfig();
   2. request.getServletConfig();
   3. response.getServletConfig();
   4. getServletContext().getServletConfig();
   5. request.getSession().getServletConfig();

1. How can a servlet access the application’s ServletContext object? (3 correct answers)
   1. getServletContext();
   2. request.getServletContext();
   3. response.getServletContext();
   4. getServletConfig().getServletContext();
   5. request.getSession().getServletContext();

1. The value of a servlet init parameter can be changed programmatically, but the value of a context init parameter cannot. (1 correct answer)
   1. true
   2. false

1. A context init parameter cannot have the same name with a servlet init parameter. (1 correct answer)
   1. true
   2. false

1. A servlet init parameter cannot have the same name with the servlet it refers to. (1 correct answer)
   1. true
   2. false

1. Where is a servlet init parameter stored after the servlet is initialized and available for use? (1 correct answer)
   1. In the ServletConfig object of the servlet.
   2. In the ServletContext object of the web application.

1. Where is a context init parameter stored after the servlet is initialized and available for use? (1 correct answer)
   1. In the ServletConfig object of the servlet.
   2. In the ServletContext object of the web application.

1. Assume the servlet HelloServlet that belongs to packagecom. The file HelloServlet.class is placed in the directory WEB-INF/classes/com. Is this a correct declaration of an init parameter for this servlet? (1 correct answer)

|  |
| --- |
| <**servlet**>      <**servlet-name**>Hello Servlet</**servlet-name**>      <**servlet-class**>classes.com.HelloServlet</**servlet-class**>      <**init-param**>          <**param-name**>this</**param-name**>          <**param-value**>Hello!</**param-value**>      </**init-param**>  </**servlet**> |

* 1. Yes.
  2. No, because servlet-name contains a space.
  3. No, because servlet-class has a wrong value.
  4. No, because param-name is a reserved Java keyword.
  5. No, because param-value contains an explanation mark (!).
  6. No, because init-param should be inside a servlet-mapping element.

1. A context init parameter with name “ice” and value “tea” is properly declared. What is the result of this code? (1 correct answer)

|  |
| --- |
| **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(          getServletContext().getInitParameter("tea")      );  } |

* 1. A NullPointerException occurs at runtime.
  2. The browser displays an empty page.
  3. The browser displays “null”.
  4. The browser displays “ice”.
  5. The browser displays “tea”.
  6. Compilation fails.

1. A context init parameter with name “ice” and value “tea” is properly declared. What is the result of this code? (1 correct answer)

|  |
| --- |
| **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(getContextParameter("ice"));  } |

* 1. A NullPointerException occurs at runtime.
  2. The browser displays an empty page.
  3. The browser displays “null”.
  4. The browser displays “ice”.
  5. The browser displays “tea”.
  6. Compilation fails.

1. The Container creates a single instance for every servlet. The client requests are served with various threads. (1 correct answer)
   1. true
   2. false

1. Assume that the container is running on port 9999 of localhost, our application is called test1 and that the following servlet is declared with url pattern /processor. What is the output? (1 correct answer)

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(request.getContextPath());  }  } |

* 1. <http://localhost:9999/test1/processor>
  2. /test1/processor
  3. /processor
  4. /test1

1. Assume that the container is running on port 9999 of localhost, our application is called test1 and that the following servlet is declared with url pattern /processor. What is the output? (1 correct answer)

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(request.getServletPath());  }  } |

* 1. <http://localhost:9999/test1/processor>
  2. /test1/processor
  3. /processor
  4. /test1

1. Assume that the container is running on port 9999 of localhost, our application is called test1 and that the following servlet is declared with url pattern /processor. What is the output? (1 correct answer)

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(request.getRequestURL());  }  } |

* 1. <http://localhost:9999/test1/processor>
  2. /test1/processor
  3. /processor
  4. /test1

1. Assume that the container is running on port 9999 of localhost, our application is called test1 and that the following servlet is declared with url pattern /processor. What is the output? (1 correct answer)

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(request.getRequestURI());  }  } |

* 1. <http://localhost:9999/test1/processor>
  2. /test1/processor
  3. /processor
  4. /test1

1. Consider this form.

|  |
| --- |
| <**form** method="post" action="processor">     <**input** type="submit" value="OK"/>  </**form**> |

And this servlet.

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(request.getMethod());  }  } |

What is the output when the form is submitted? (1 correct answer)

* 1. GET
  2. POST
  3. A HTTP 405 message informs that GET is not supported.
  4. A HTTP 405 message informs that POST is not supported.

1. Consider this form.

|  |
| --- |
| <**form** method="post" action="processor">     <**input** type="submit" value="OK"/>  </**form**> |

And this servlet.

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().println(request.getMethod());  }  **public** **void** doPost(HttpServletResponse response,                    HttpServletRequest request)  **throws** ServletException, IOException {      doGet(request, response);  }  } |

What is the output when the form is submitted? (1 correct answer)

* 1. GET
  2. POST
  3. A HTTP 405 message informs that GET is not supported.
  4. A HTTP 405 message informs that POST is not supported.

1. Consider this form.

|  |
| --- |
| <**form** name="form" method="post" action="processor">     <**input** name="submit" type="submit" value="OK"/>  </**form**> |

And this servlet.

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doPost(HttpServletRequest request,                     HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().format("%s %s",      request.getParameter("form"),      request.getParameter("name"));  }  } |

What is the output when the form is submitted? (1 correct answer)

* 1. form OK
  2. null OK
  3. form null
  4. null null

1. Consider this form.

|  |
| --- |
| <**form** name="form" method="post" action="processor">     <**input** name="submit" type="submit" value="OK"/>  </**form**> |

And this servlet.

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doPost(HttpServletRequest request,                     HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().format("%s %s",      request.getParameter("form"),      request.getParameter("submit"));  }  } |

What is the output when the form is submitted? (1 correct answer)

* 1. form OK
  2. null OK
  3. form null
  4. null null

1. Consider this form.

|  |
| --- |
| <**form** name="form" method="post" action="processor">     <**input** type="text" name="name" value="OK"/>     <**input** name="submit" type="submit" value="OK"/>  </**form**> |

And this servlet.

|  |
| --- |
| **public** **class** Processor **extends** HttpServlet {  **public** **void** doPost(HttpServletRequest request,                     HttpServletResponse response)  **throws** ServletException, IOException {      response.getWriter().format("%s %s",      request.getParameter("name"),      request.getParameter("OK"));  }  } |

What is the output when the form is submitted? (1 correct answer)

* 1. OK OK
  2. null OK
  3. OK null
  4. null null

1. The following types are ALL interfaces and are used as listeners in a web application. (1 correct answer)  
   → ServletContextListener  
   → ServletContextAttributeListener  
   → ServletRequestListener  
   → ServletRequestAttributeListener  
   → HttpSessionListener  
   → HttpSessionBindingListener  
   → HttpSessionAttributeListener  
   → HttpSessionActivationListener
   1. true
   2. false

1. Consider the interface ServletContextListener. What is the signature of the method that is invoked when the servlet context is about to be shut down? (1 correct answer)
   1. void contextDeleted(ServletContextEvent)
   2. void contextDestroyed(ServletContextEvent)
   3. void servletContextDeleted(ServletContextEvent)
   4. void servletContextDestroyed(ServletContextEvent)

1. There is a properly declared HttpSessionAttributeListener. How many times its attributeRemovedmethod is invoked when the following servlet is accessed once? (1 correct answer)

|  |
| --- |
| **public** **class** Test **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      request.getSession().setAttribute("planet", "earth");      request.getSession().removeAttribute("planet");      request.getSession().removeAttribute("planet");  }  } |

* 1. 0
  2. 1
  3. 2

1. There is a properly declared HttpSessionAttributeListener. How many times its attributeRemovedmethod is invoked when the following servlet is accessed once? (1 correct answer)

|  |
| --- |
| **public** **class** Test **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      request.getSession().setAttribute("planet", "earth");      request.getSession().invalidate();      request.getSession().removeAttribute("planet");      request.getSession().setAttribute("planet", "earth");  }  } |

* 1. 0
  2. 1
  3. 2

1. There is a properly declared HttpSessionAttributeListener. How many times its attributeReplacedmethod is invoked when the following servlet is accessed once? (1 correct answer)

|  |
| --- |
| **public** **class** Test **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      request.getSession().setAttribute("planet", "earth");      request.getSession().removeAttribute("earth");      request.getSession().setAttribute("planet", "earth");  }  } |

* 1. 0
  2. 1
  3. 2

1. There is a properly declared HttpSessionAttributeListener. How many times its attributeReplacedmethod is invoked when the following servlet is accessed once? (1 correct answer)

|  |
| --- |
| **public** **class** Test **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request,                    HttpServletResponse response)  **throws** ServletException, IOException {      request.getSession().setAttribute("planet", "earth");      request.getSession().removeAttribute("planet");      request.getSession().setAttribute("planet", "venus");      request.setAttribute("planet", "earth");      request.setAttribute("planet", "venus");  }  } |

* 1. 0
  2. 1
  3. 2