Java  **Web Applications** are used to create dynamic websites. Java provides support for web application through  **Servlets** and **JSPs.**If you are going to create a java web application then i hope below points will be useful for you.Java Web Application best practices.

1. Use MVC Architecture –  Model-View-Controller (MVC) is fundamental to the design of good Java EE applications.
2. Design architecture based on the requirements like Whether application only support Web Client or Mobile Client also.Write the code based on that.
3. Use Spring Framework.
4. Apply Security in your application – Spring Security or OAuth.
5. Go for SOA Architecture –  Imagine you are developing a web-application and you decide to decouple the functionality from the presentation of the application, because it affords greater freedom.
6. Use SPA(Single Page Application) –  SPA, can do almost everything that MPA does AND it loads much faster as well.
7. Use  Interfaces instead of Abstract classes.
8. Instead of writing your own from scratch,use standard library.
9. Secure User password using Java MD5 Hashing and Salting.
10. Use Property File for Configuration – Don’t Hard Code configuration related information in your code. Also  Keep Your User Interface labels in the properties files.
11. Use Log4j for Application logging. – Keep your logging level (DEBUG,INFO ..) based on the environment.
12. Configure  JNDI Datasources.
13. Apply  Caching like Hibernate Second level cache at required places to improve the performance.
14. Use JSON (JavaScript Object Notation) is a lightweight format that is used for data interchanging.
15. Try Hudson/Jenkins for Automated Build Process.
16. Try Maven or Gradle to build the application. And get the advantages of maven plugins.
17. Use String Carefully in your code.
18. Don’t  start your own thread in a container managed application, like a web application or an enterprise (EJB) application.
19. Handle HttpSession Carefully – Invalidate the HttpSession after certain inactive period time.Also  Session sharing between two war file is not possible. Don’t declare HttpSession as instance variable.
20. Return Empty Collection instead of Null.
21. Decouple your components and  Generally think about the separation of concerns.
22. Create User defined Exception wherever required.  Converting Checked Exception into RuntimeException –  This Java best practice provides benefits, in terms of   restricting specific exception into specific modules, like SQLException into DAO layer and throwing meaningful Runtime Exception to client layer.
23. Handle All the Exceptions Carefully.
24. Write Unit Test cases and use SonarQube with PMD tool for Unit test Coverage and coding issues.
25. Configure SSL if required and get certification authority .
26. Use Table Pagination Properly.
27. Minify your Html and CSS Codes to improve the performance.
28. Use layered Structure in UI side al well as back end or server side.
29. Try Velocity or Freemarker. These are all templating engines that allows the separation of UI logic and business logic, thus facilitating changes in the presentation while minimizing the changes on the business side.
30. Use Apache common utility classes.
31. Choose Server side session or Client Side Session carefully.
32. Check your application with  Pen Testing tool to  ensure the security of websites.
33. Try  Websockets –  This  provides an alternative to this (Push” or Comet techniques) limitation by providing bi-directional, full-duplex, real-time, client/server communications
34. Use Connection Pooling and transaction management.
35. Don’t waiting for Mail Send Response. Send Mail Asynchronously or schedule it.
36. Test your application with JMeter for Load test.
37. Try these [Useful Java Tools](http://www.techpages.org/java/java-development-tools/2487/).
38. Use Eclipse Plugins based on your server and test the application locally.
39. Choose right design patters at required places.
40. Create Java Web Application as Maven multi module project.
41. Check with Profiling tools like JProfiler to identify memory leaks.
42. Configure Jamon in your application –  The Java Application Monitor (JAMon) is a free, simple, high performance, thread safe, Java API that allows developers to easily monitor production applications.
43. Configure Your Web Application based on the environments because Production environment configuration differ from Development environment.
44. Use Cross cutting concerns  like Spring AOP concepts  if required.
45. Maintain Release/build Number for ever releases.

I hope above tips really useful for new Java developers who is going start new web application. Please share your comments through below comment section and i will add it in the above list