JAR vs WAR vs EAR

|  |  |  |
| --- | --- | --- |
| JAR | WAR | EAR |
| EJB modules which contains enterprise  java beans class files and EJB deployment  descriptor are packed as JAR files with  .jar extenstion | Web modules which contains Servlet class files,JSP FIles,supporting files, GIF and HTML files are packaged as JAR file with .war (web archive) extension | All jar and .war are packaged as JAR file with .ear (enterprise archive) extension and deployed into Application Server. |
| <http://stackoverflow.com/questions/5936734/why-should-we-use-ejb>  Use EJB if you wish to take advantage of the following services provided by the Java Enterprise Edition (Java EE) server:   1. Clustering 2. Concurrency 3. Connection Pool to Database 4. Security 5. Schedule service | A JSP is compiled to a servlet the first time it is run. That means that there's no real runtime difference between them.  However, most have a tradition to use servlets for controllers and JSPs for views. Since controllers are just java classes you can get full tool support (code completion etc.) |  |

Javascript Ajax

GET:

Var xmlhttp=new XMLHttpRequest();

xmlhttp.open("GET","demo\_get.asp?t=" + Math.random(),true);

xmlhttp.send();

POST:

var http = new XMLHttpRequest();

var url = "get\_data.php";

var params = "lorem=ipsum&name=binny";

http.open("POST", url, true);

Javascript Events

**Event.preventDefault**

The preventDefault method prevents an event from carrying out its default functionality. For example, you would use preventDefault on an A element to stop clicking that element from leaving the current page:

//clicking the link will \*not\* allow the user to leave the page

myChildElement.onclick = function(e) {

e.preventDefault();

console.log('brick me!');

};

//clicking the parent node will run the following console statement because event propagation occurs

logo.parentNode.onclick = function(e) {

console.log('you bricked my child!');

};

**Event.stopPropagation**

The second method, stopPropagation, allows the event's default functionality to happen but prevents the event from propagating:

//clicking the element will allow the default action to occur but propagation will be stopped...

myChildElement.onclick = function(e) {

e.stopPropagation();

console.log('prop stop! no bubbles!');

};

//since propagation was stopped by the child element's onClick, this message will never be seen!

myChildElement.parentNode.onclick = function(e) {

console.log('you will never see this message!');

};

Ref:

<http://stackoverflow.com/questions/5963669/whats-the-difference-between-event-stoppropagation-and-event-preventdefault>