Hidden Fields and Cookies

Web is stateless.Both Server and browser process new URL request independent of previous requests. In other word, browser and server do not perform passing of information from one page to another page by default.

There are situation in which you want the processing of one web page to be dependent on the processing pervious page .For example you may want to create a form that collect general information about the user, such as name address and link it to subsequent forms to collect more information. However those forms will vary depending on what country the user has entered in the first form.

A number of capabilities have been successively introduced to enable web application to be built upon the stateless design of the web. Hidden form field were introduced first followed by cookies.

Hidden Form Field

Hidden form field are the text field that are not displayed and cannot modified by a user.

A user-program sets hidden fields to a particular value when the server sends form to a browser. When a user fills out and submits s a form containing a hidden field, the value originally stored in the field returns to server. The server uses the information stored in the hidden field to maintain state information about the browser.

Example for hidden field:

```
<html>
<body>
<script language="javascript">
 function Test()
 {
   document.frm.browinfo.value =navigator.appName;

}

</script>
<form name="frm" action="mailto:urmail@hotmail.com" method=get>
<br>
&nbsp;&nbsp;*&nbsp;&nbsp;Name:<input type=text  name="name">
&nbsp;&nbsp<input type=hidden  name="browinfo">
<input type=submit value="submit" onClick="Test()">
</form>
</body></html>
```

In the above example you the browser sends the information about the browser that you are using into you mail addresss. Usually Hidden fields are useful cgi pages when the information from one page needs to be sending to the page.

Cookies

Hidden for field were introduced to enable CGI program to maintain state information about web Browser. They work well in situation where the state information is to be maintained for a short period of time as is the case when the user fills out a series of

forms. However hidden fields do not allow state information to be used within a single browser session. When a user exits the browser, the information contained in a hidden form field is lost forever.
Netscape developed the cookie as a mean to store state related and other information in a persistent manner. The information stored state-related and other information in a persistent manner. The information stored in a cookie maintain between browser sessions. It survives when the user turns off his or her machine. Cookies allow CGI and other program to store information on the web borwser for significantlylo0onger time period.Netscape navigator, internet explorer and most popular browser supports cookies.
Cookies are the chunk of information that is written by the web site to the visitor's computers hard drive. A cookie consists of information sent by a server –side programming in response to a URL request by the browser. The browser stored the information in the local cookie file. Different browser stores cookie in different file Eg: Netscape navigator stores cookies in a file named cookie.txt. It stores coolies in multiple files in the \window\cookie directory.
When a browser request a URL forms a web server the browser first search the local cookies files to see if the URL of any of its cookies matches the URL that it is requesting. The browser then sends the web server, as part of the URL request, the information contained in the matching cookie or cookies...
When the first time this idea was conceived many rejected the virus like activity of cookie. Later, with time and usefulness of cookies it is widely accepted.
Howe is information stored in a cookie
A cookie is created when a CGI program includes a set-cookie headers part of an HTTP response. This response is generated when a browser request the URL of the CGI program. The syntax of the set-cookie header is:

```
Set-Cookie: NAME=Value; [EXPIRES=date;] [PATH=pathname;]
 [DOMAIN=domainname;] [SECURE]
```

name=value

This string is a sequence of characters excluding semicolons, commas, and white space. If there is a need to place such data in the name or value, some encoding method such as URL style `%XX` encoding is recommended, though no encoding is defined or required. This is the only required attribute on the `Set-Cookie` header.

expires=*date*

The `expires` attribute specifies a date string that defines the valid lifetime of that cookie. Once the expiration date has been reached, the cookie will no longer be stored or given out. The date string is formatted as:

```
Wdy, DD-Mon-YYYY HH:MM:SS GMT
```

`expires` is an optional attribute. If not specified, the cookie will expire when the user's session ends.

NOTE:

There is a bug in Netscape Navigator version 1.1 and earlier. Only cookies whose `path` attribute is set explicitly to "/" will be properly saved between sessions if they have an `expires` attribute.

`domain=`*`domain_name`*

When searching the cookie list for valid cookies, a comparison of the `domain` attributes of the cookie is made with the Internet domain name of the host from which the URL will be fetched. If there is a tail match, then the cookie will go through `path` matching to see if it should be sent. "Tail matching" means that `domain` attribute is matched against the tail of the fully qualified domain name of the host. A `domain` attribute of *acme.com* would match host names *anvil.acme.com* as well as *shipping.crate.acme.com*.

The default value of `domain` is the host name of the server which generated the cookie response. `domain=`*`domain_name is also optional`*

`path=`*`path`*

The `path` attribute is used to specify the subset of URLs in a domain for which the cookie is valid. If a cookie has already passed `domain` matching, then the pathname component of the URL is compared with the path attribute, and if there is a match, the cookie is considered valid and is sent along with the URL request. The path */foo* would match */foobar* and */foo/bar.html*. The path / is the most general path.

If the `path` is not specified, it as assumed to be the same path as the document being described by the header which contains the cookie. `path=`*`path is also optional`*

`secure`

If a cookie is marked `secure`, it will only be transmitted if the communications channel with the host is a secure one. Currently this means that secure cookies will only be sent to HTTPS (HTTP over SSL) servers.

If `secure` is not specified, a cookie is considered safe to be sent in the clear over unsecured channels. Secure is alos optional

When a request for cookie information is made, the list of cookie information is searched for all URLs which match the current URL. Any matches are returned in this format:
cookie: NAME1=*string1*; NAME2=*string2*; ...

Using JavsScript with Cookies

Java Script can take full advantage of cookie by reading and setting then locally on the browser eliminating the need for the cookies to be proceeding by CGI programs. A JavaScript can then forwarded any information the CGI program requires for the processing.The cookie associated with a document is set/retrieved using the document cookie property

Eg:

```
Var email,expirationDate
Email=”info@nec.edu.np;
expirationDate=“Thursday, 21-may-06 12:00:00 GMT“;
document,cookie=“email=“+email+“;expires=“+expirationDate;
```

This statemetn set the values of the cookie property of the current document to the string “email=info@nec.edu.np;expires= Thursday, 21-may-06 12:00:00 GMT” Note that the expires field is required to kkep the cooki from expiring after the current browsersession

The value of cookie is retrieved by using the statement

```
Var retrieved_cooki_value=document.cookie;
```

The retrieved_cooki_value variable will be asigned the value email=info@nec.edu.npif multiple cookie s had been ser for the document ,then the retrieved_cooki_valuewould contain a list of semicolon-seperated name=value pairs;

Example:

```
<html>
<body>
<script language="javascript">
var email,ed;
ed="Monday,21-June-05 12:00:00 GMT";
email="aakash@hotmail.com";
document.cookie="email="+email+";expires="+ed;
document.cookie="name='aakash';expires='tuesday, 22-June-04 12:00:00 GMT";
document.cookie="test='test';expires='tuesday, 22-June-04 12:00:00 
GMT;path='web\';Secure";
var len=document.cookie;
document.write(len);
</script>
</body></html>
```

Comparison

Tradeoff	**Cookies**	**Hidden Form**
Ease of use	Requires cookie string Passing	Requires Form set up and access
Browser Support	Navigator ,IE etc	Almost all browser
Server Support	May not e supported	Supported by all browser
Performance	Slower(requires disk I/O)	Faster (implemented in Ram)
Persistence Storage	Supported	Not supported