ACCP I7.1 – SEMESTER 3

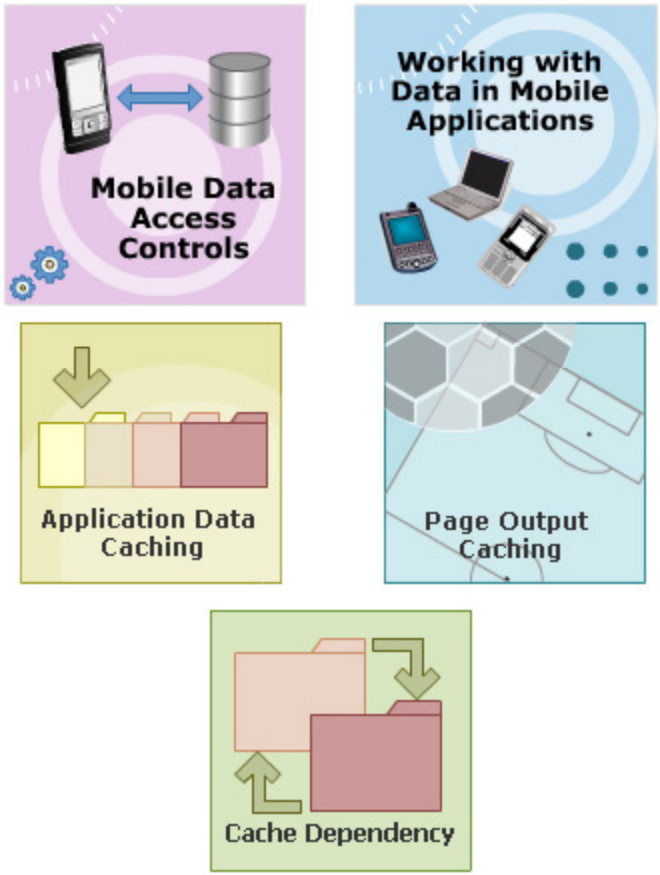
DATABASE HANDLING IN ASP.NET

Module 5 & 6

Session 3 – Caching and Security

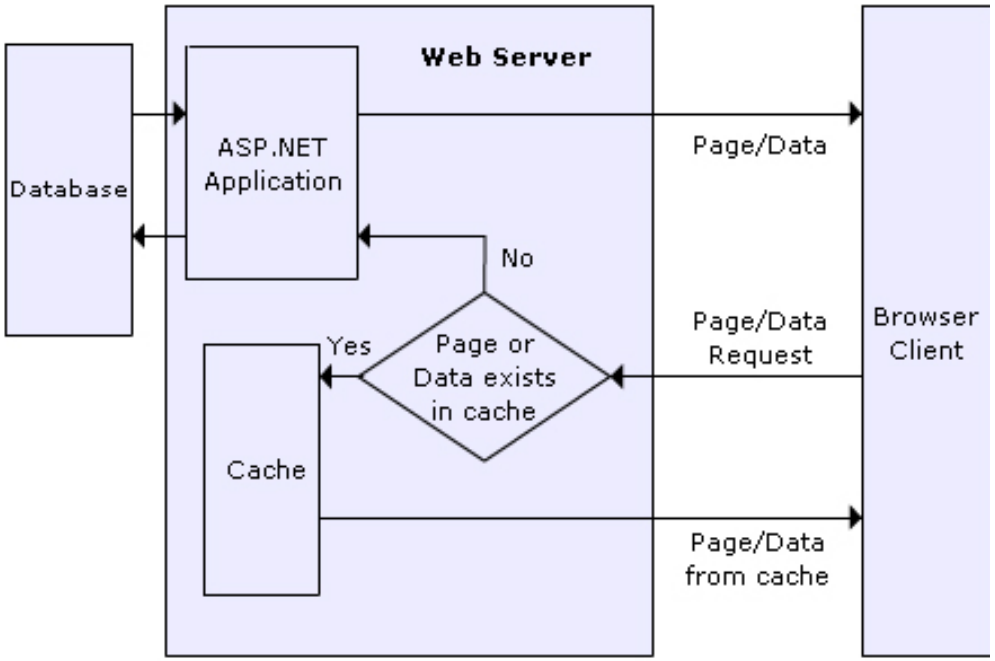
Objectives

- Introduce Caching in ASP.NET
- Discuss Application Data Caching
- Discuss Page Output Caching
- Discuss Cache Dependency
- Introduce Mobile Data Access Controls



ASP.NET Caching Concept

- ASP.NET application's performance and efficiency can be improved by maintaining a temporary local store called a cache.
- This cache would store frequently used application data or pages

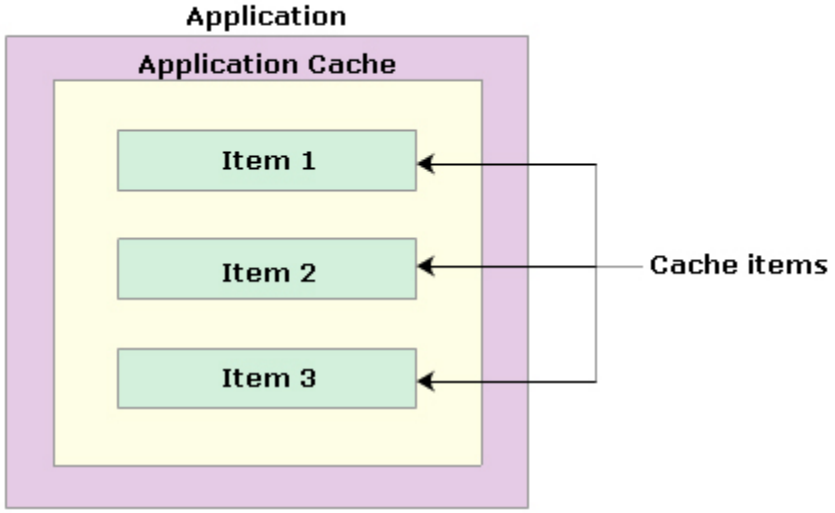


Caching types

- Two kinds: Data Cache and Page Cache
- Data Cache allows caching of any .NET object
  - Often used to limit trips to database but still give dynamic look and feel to web site
- Page Cache allows caching of an ASPX or ASCX
  - Used to produce high performance but seldom changing pages

Application Data Caching

- Application caching is a mechanism in ASP.NET to persist the frequently used application objects or data for a Web application
- System.Web.Caching.Cache is used for caching



Adding Items to Cache

- Key/Value Pairs
- Cache.Insert(string, Object) Method
- Cache.Insert(string, Object, CacheDependency, DateTime, TimeSpan) Method

```
string t = Cache["TimeString"] as
string;
if (t == null)
{
        t = DateTime.Now.ToString();
        Cache.Insert("TimeString", t);
}
_lblTime.Text = t;
```

Retrieving and Deleting Cache Items

Retrieving Cache Items:

```
dtPatients = Cache["PatientsDataTable"];
```

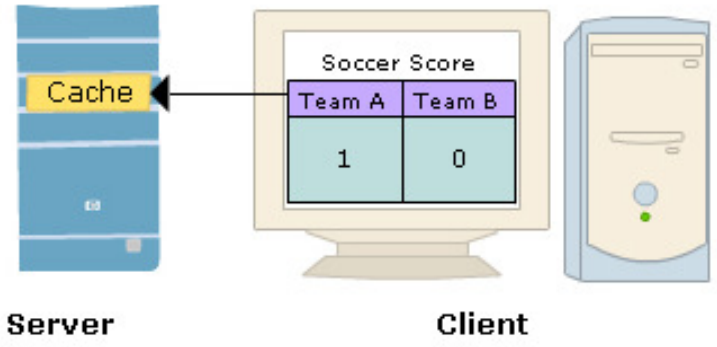
Deleting Cache Items:

```
Cache.Remove("PatientsDataTable");
```

Declarative Page Output Caching

@OutputCache Directive

```
<%@ OutputCache Duration="#ofseconds"
Location="Any | Client | Downstream | Server |
None"
Shared="True | False"
VaryByControl="controlname"
VaryByCustom="browser | customstring"
VaryByHeader="headers"
VaryByParam="parametername" %>
```



Fragment Caching

- Fragment Caching: ***caching portions of a page***
- There are times when certain sections of the page need to be cached because for most of time they remain static.
- Examples: copy right information on the Web pages, weekly summaries on a stock information Web page, logo or punch line of an organization displayed across pages on a Website

Cache Dependency

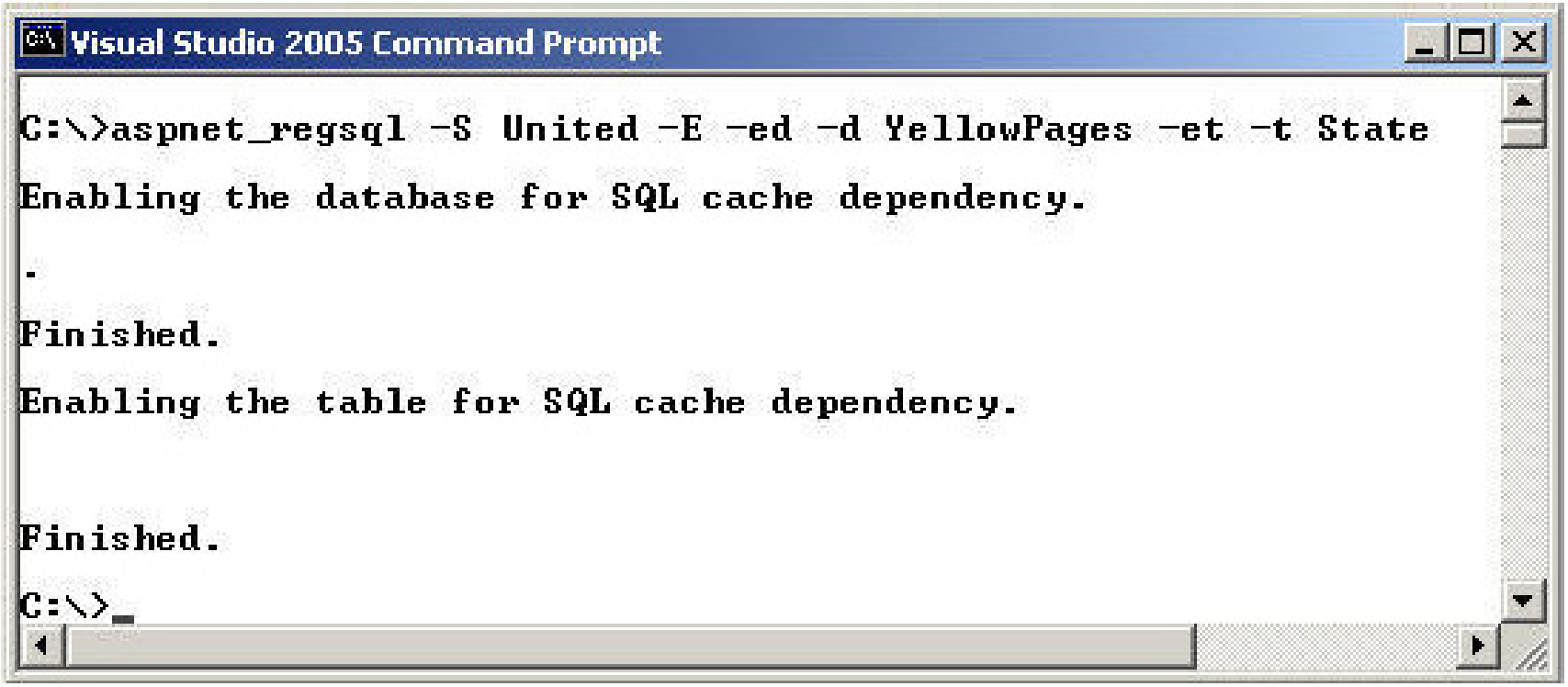
- **Invalidation:** the process of removing data from the cache so that the cache can be refreshed with the latest data
- Invalidating a cache item is made possible in ASP.NET by making the cache item dependent
- Type of dependencies:
  - File-based

SQL Cache Dependency

- 3 things have to be setup
  - SQL Server
    - Run utility or wait for SQL Server 2005
  - Application
    - Web.Config
  - Your Code
    - Could be declaratively through the OutputCache directive
    - Could be in code

SQL Cache Dependency

aspnet_regsql utility



SQL Cache Dependency

The Web.config file

```
<caching>
      <sqlCacheDependency enabled="true"
                          pollTime="1000">
            <databases>
                  <add name="Northwind“

connectionStringName="NorthwindConnectionString1"/>
            </databases>
      </sqlCacheDependency>
</caching>
```

Cache Dependency

In Page Output Caching

- Declarative Cache Dependency

```
<%@ OutputCache Duration="60"
VaryByParam="None"
SqlDependency="YellowPages:State" %>
```

- Programmatic Cache Dependency

```
...
Response.Cache.SetCacheability(HttpCacheability
.Public);
Response.Cache.SetExpires(DateTime.Now.AddSecon
ds(60));

SqlCacheDependency scdYellowPages = new
SqlCacheDependency("YellowPages", "State");

Response.AddCacheDependency(scdYellowPages);
...
```

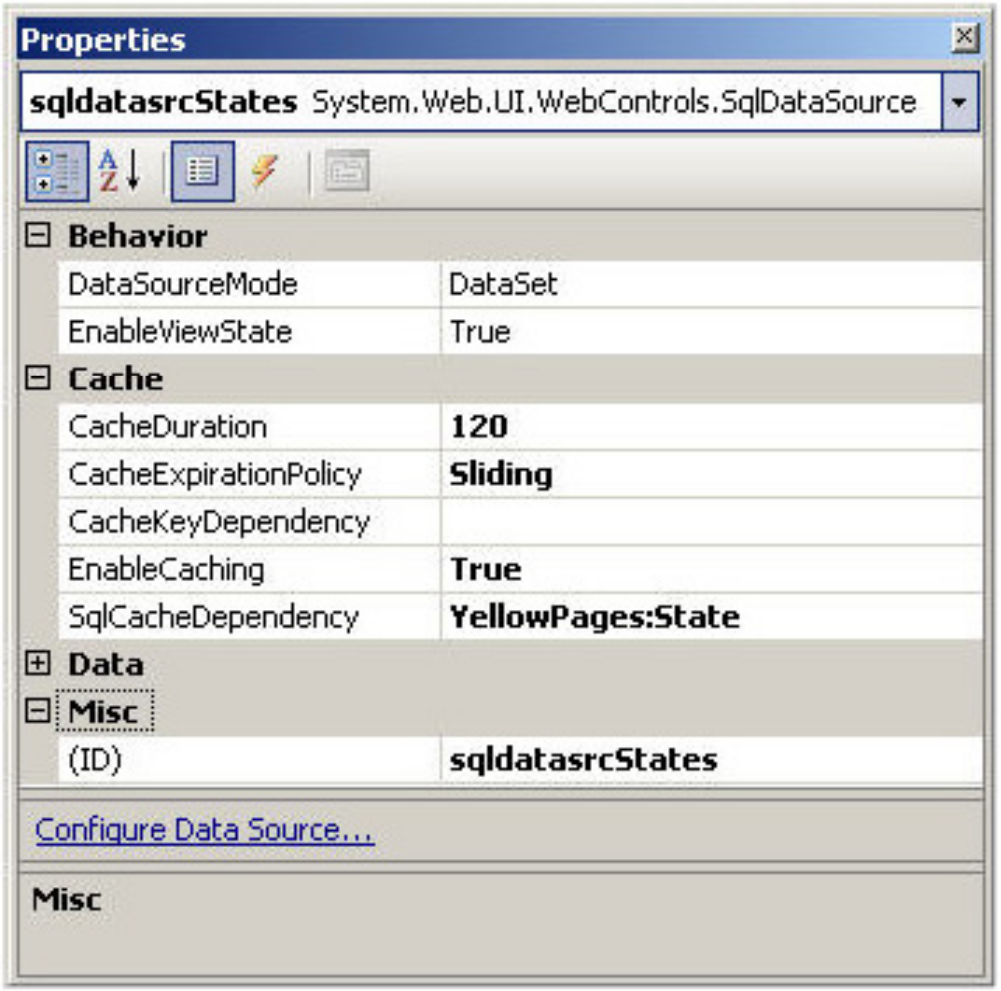
Cache Dependency

In Application Data Caching

```
DataSet1 dsetState = new DataSet1();

DataSet1TableAdapters.StateTableAdapter taState =
  new DataSet1TableAdapters.StateTableAdapter();

DataSet1.StateDataTable dtState = dsetState.State;
taState.Fill(dtState);
...

// Step 1
SqlCacheDependency scdYellowPages = new
SqlCacheDependency("YellowPages", "State");

// Step 2
if (Cache["StateDataTable"] == null)
{
  // Step 3
  Cache.Insert("StateDataTable", dtState,
scdYellowPages, DateTime.Now.AddSeconds(60),
TimeSpan.Zero);
}
// Step 4
gvwState.DataSource = Cache["StateDataTable"];
gvwState.DataBind();
```

Cache Dependency

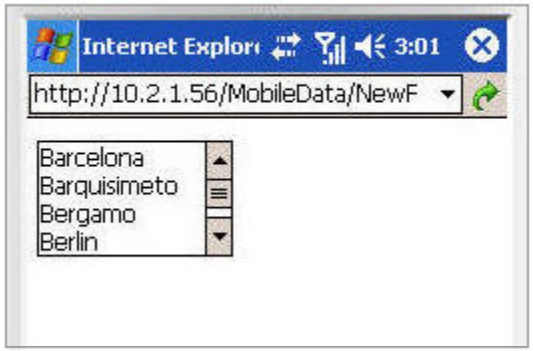
In Data Source Controls



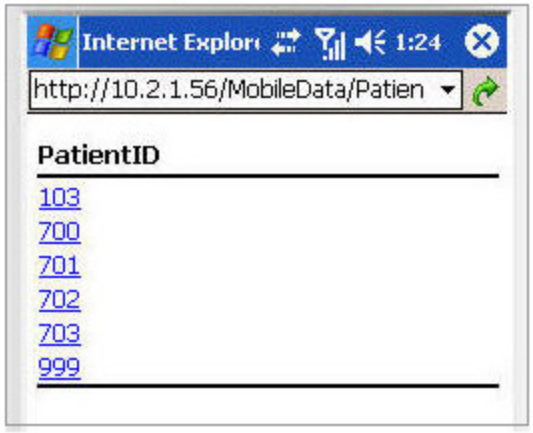
Mobile Controls Supporting Data Access

- ASP.NET provides number of controls as part of the ***System.Web.UI.MobileControls*** namespace to work with data in mobile Web applications.
- Commonly used mobile controls are
  - SelectionList
  - List
  - ObjectList

SelectionList Control



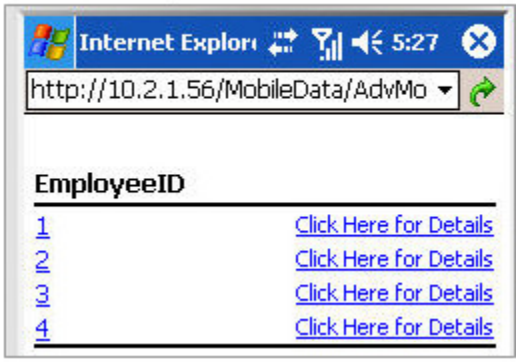
ObjectList Control



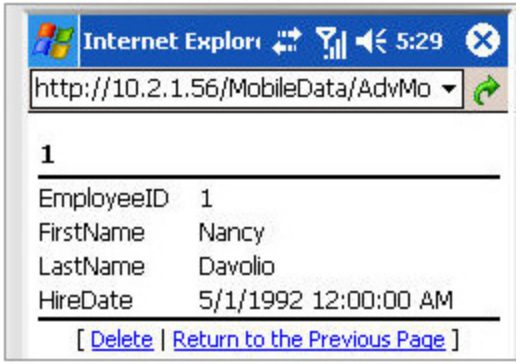
Binding the "ObjectList" Control

```
protected void slstCities_SelectedIndexChanged
(object sender, EventArgs e)
{
    // Retrieve records from the table based on
    selected city.
    . . .
    // Step 1
    daCities.Fill(dsetCities);

    // Step 2 - Populate the records into the
    ObjectList
    this.objlstEmp.DataSource = dsetCities;

    // Step 3
    this.objlstEmp.DataMember =
    dsetCities.Tables[0];

    // Step 4 - Configure the properties of the
    ObjectList
    this.objlstEmp.LabelField = "EmployeeID";

    //Step 5
    this.objlstEmp.MoreText = "Click Here for
    Details";
    this.objlstEmp.BackCommandText = "Return to
    the Previous Page";

    //Step 6
    this.objlstEmp.DataBind();
}
```

ObjectList Control



Detail view of ObjectList

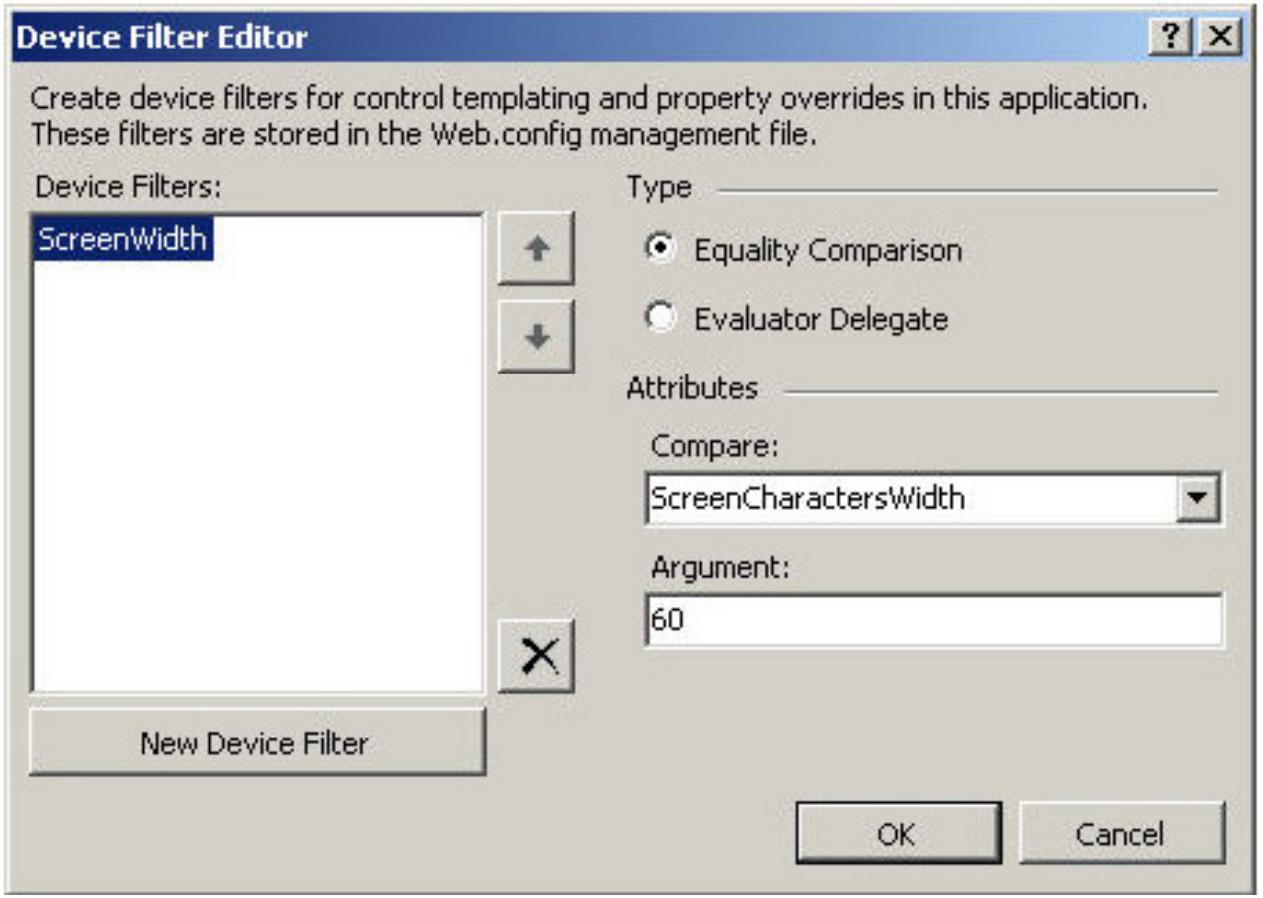


The ObjectList Control

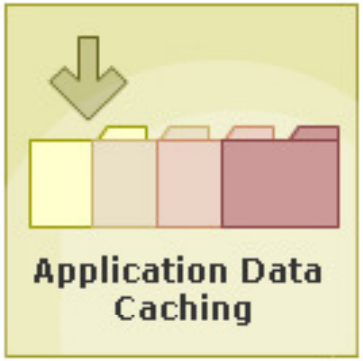
Device Filters

- Device filters are the means using which you can customize your mobile Web application output for specific mobile devices or device capabilities

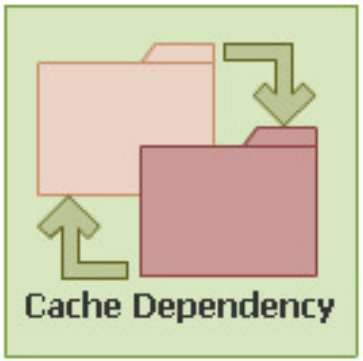
Device Filters



Summary


Application Data Caching


Page Output Caching


Cache Dependency


Mobile Data Access Controls


Working with Data in Mobile Applications