



# Citizen Science

---

Data Collection Using Volunteers  
and ASP.NET to Collect Turkey  
Gobbling Data



# The Star – Meleagris gallopavo

---

- [http://video.nationalgeographic.com/video/player/animals/birds-animals/ground-birds/turkey\\_wild.html](http://video.nationalgeographic.com/video/player/animals/birds-animals/ground-birds/turkey_wild.html)



# The Questions

---

- Does latitude change in Missouri make a measurable difference in turkey gobbling activity.
- Northern Missouri (Unionville, MO – 40 degrees latitude) versus southern Missouri (West Plains, Missouri – 36 degrees latitude).
- If so, it may justify modification to the hunting regulations.



# Questions continued

---

- If not, proof that gobbling activity is so local and variable that different regulations in different parts of the state can not be justified.
- The plan is to get turkey chasers to help in trying to answers these questions.
- We decided that the web was a good and economically feasible way of conducting a turkey gobbling study.



# The Plan

---

- Get 500 Gobbleteers to sign up to collect turkey gobbling data starting in the spring of 2007.
- <http://www.mdc.mo.gov/hunt/turkey/gobblecount.htm>
- Get them to use a web-based application to enter their data.
- <http://mdc4.mdc.mo.gov/applications/Gobbleteer/Login.aspx>
- We would analyze this data to answer our turkey gobbling questions.



# The Overall Design

---

- Data stored in a 9i Oracle database.
- Direct access through stored procedures found in a Oracle package.
- Web services created using the Microsoft and the .NET model.
- Web services call stored procedures.
- UI is through the browser using forms and tables written in ASP.NET pages.



# Database Facts

---

- All PL/SQL code stored in one package and package body for easy maintenance.
- The Oracle procedures are never called directly by the application.
- The web services call the Oracle procedures directly and the application calls the web services.



# Web Services

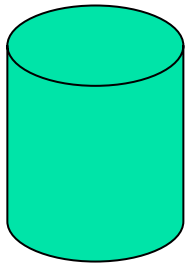
---

- Programming interface is a bit more tidy.
- Security is still an issue. If your services access your systems, you have created a vulnerability.
- <http://www.google.com/search?hl=en&q=hacking+web+services&btnG=Google+Search>
- <http://www.xml.com/pub/a/ws/2003/03/04/security.html?page=2>

# Database Access

IIS server with  
.NET pages and  
web services

Oracle 9i with  
PL/SQL code



Gobbleteer





# The Browser Interface

---

- XHTML 1.1 Strict compliance with provides a standard look and feel to all MO Dept. of Conservation web applications across different browsers.
- Class XHTMLStrictFilter enforces XHTML compliance for cases where VS 2003 and .NET 1.1 don't.
- Class SecureQueryString handles encrypting query string across requests.



# Benefits to HTML standards

---

- Greater number of users are accommodated.
- Search engine spiders can read the search engine optimized content.
- Website will be more accessible.
- Building sites with standards compliant coding will degrade/last longer as the standards are upgraded. When a new standard is implemented it will take less work to upgrade.
- Lower cost of producing the website.
- Website will be backwards compliant as the browsers improve.
- Easier to maintain and update content.
- Website will be compatible with the current browsers and future browsers.
- Faster download time.
- Page(s) will be rendered faster.
- Page(s) will render better.
- The browsers are catching up with the standards.



# Visual Studio 2003 and IIS non-compliant behavior

---

- There are several known cases of html rendering by the IIS server that causes non-complaint html.
- This is getting better in newer versions of IIS and VS as Microsoft begins to conform to these web standards.
- Form, input, script tags are all cases where the rendering of the html causes the resulting html to be non-compliant with the XHTML 1.1 Strict standard.
- Class XHTMLStrictFilter finds and filters this “non-compliant” code and replaces it with “compliant” html at the beginning of each request.



# Class XHTMLStrictFilter

---

- [Gobbleteer\Global.asax.vb](#)
- [Gobbleteer\incEEOclasses.vb](#)



# Example of XHTMLStrictFilter

---

## **IMPORTANT CODE SNIPPETS**

```
Private Shared _regViewState As New Regex("<input type=""hidden""  
name=""__VIEWSTATE"" value=""*.*)" />", RegexOptions.Compiled Or  
RegexOptions.IgnoreCase)
```

```
If Not _viewStateFix And _regViewState.IsMatch(htmlOutput) Then
```

```
    Dim viewStateMatch As String = _regViewState.Match(htmlOutput).Value
```

```
    htmlOutput = Regex.Replace(htmlOutput, Regex.Escape(viewStateMatch),  
    "<div>" + viewStateMatch + "</div>", RegexOptions.IgnoreCase)  
    _viewStateFix = True
```

```
End If
```

```
...
```

## **RESULTS**

<http://mdc4.mdc.mo.gov/applications/Gobbleteer/Login.aspx> - proper XHTML 1.1 Strict

<http://www.htmlhelp.org/tools/validator/direct.html.en>



# Maintaining data across page requests

---

- Options: Cookies, Query strings, Viewstate, Session State, Application State
- Cookies can be disabled on the browser.
- Session state and application state can be good choices, but we try to avoid these options on our server configurations.
- Viewstate can get very large in the page and slow down loading times on the browser.
- For our purposes, a semi-secure query string works well.
- Class SecureQueryString handles all encoding and decoding of the query string.



# Class SecureQueryString

---

- [Gobbleteer\incEEOclasses.vb](#)
- [Gobbleteer\GobblingData.aspx.vb](#)



# Example of SecureQueryString

---

## **IMPORTANT CODE SNIPPETS**

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)  
Handles MyBase.Load
```

```
    Dim qs_in As SecureQueryString  
    Dim temp_pk As String ' gob_id  
    Dim temp_key As String ' gob_user  
    Dim temp_cnty As String ' county_id
```

Try

```
    qs_in = New SecureQueryString(Request.QueryString("vars"))  
    temp_pk = qs_in("PK") ' gob_id  
    temp_key = qs_in("KEY") 'gob_user  
    temp_cnty = qs_in("CNTY") 'county_id
```

Catch ex As Exception

```
    Response.Redirect("http://www.mdc.mo.gov")
```

End Try

...

# Example of SecureQueryString

## Continued

### IMPORTANT CODE SNIPPETS

Private Sub hypersumall\_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles hypersumall.Load

```
Dim qs_in As SecureQueryString
Dim url As String
Dim query_string As String
Dim temp_pk As String ' gob_id
Dim temp_key As String ' gob_user
Dim temp_cnty As String 'county_id
```

Try

```
qs_in = New SecureQueryString(Request.QueryString("vars"))
url = "SummaryAll.aspx?vars=" + qs_in.ToString()
```

Catch ex As Exception

```
Response.Redirect("http://www.mdc.mo.gov")
```

End Try

...

### RESULTS

<http://mdc4.mdc.mo.gov/applications/Gobbleteer/Login.aspx>



# Conclusions

---

- 500 Gobbleteers used the system in the spring of 2007.
- 5000 data records collected.
- Jeff Beringer (MO Dept. of Conservation biologist) has been hard at work looking at the data.
- Great way of recruiting and using enthusiastic volunteers to collect data in a relatively simple manner.



# Some of the Problem Areas

---

- Is the data being collected of good quality?
- Recruiting a volunteer core is time consuming and can be labor intensive.
- Maintaining users and helping users is always a challenge.