Web Services

- A web service is a standardized way for applications to communicate over a network
- Communication is in the form of XML
- Data is sent using a protocol called SOAP (Simple Object Access Protocol)
- API is specified using WDSL (Web Services Description Language)
- There are a bunch of details in setting up communication
- We will not need to worry about these

Creating Web Services

- To create and test a web service on your local machine, you need IIS (Internet Information Services)
- File → New Web Site

Web Services in .NET

- Developing, deploying, and consuming web services in .NET is remarkably simply
- All the details for network communication (marshalling and unmarshalling data in XML) are taken care of automatically
- Creating a web service to process requests feels just like writing a normal class with methods
- Communicating with a web service feels just like invoking methods on an object – the object just happens to be remote
Creating Web Services

• Notice that each method that is exposed by the web service is labeled with `[WebMethod]`
• Compiling and running the project exposes the web service locally for testing
  • Eg, http://localhost:49324/WebService1/Service.asmx
• Navigating to this page in a browser yields a web interface for testing the web service’s methods
• You can pass parameters to methods that require it
• Invoking the methods returns the XML that gets returned in response to the query

Deploying Web Services

• Deploying a web service requires little more than copying the .asmx and .cs files onto your server
• In this class, we will not get any practice deploying web services onto remote servers

Consuming Web Services

• To connect to a web service from a .NET application, you need to add a reference to it
• In the Solution Explorer, right-click the References folder and click Add Web Reference
• In the URL box, enter the location of the web service you want to communicate with – this should end with .asmx

Consuming Web Services

[demo]
http://www.ravichugh.com/webservices/WeatherService.asmx

Consuming Web Services

• Once we’ve added our web reference, let’s say it’s called com.ravichugh.www, we can instantiate the web service and invoke its methods

```csharp
String temp =
webSvc.GetTemperatureCityState("Philadelphia", "PA");
```

• Note that there can be several types of exceptions that can be thrown when dealing with network connections
• So you should add some exception handling code
• There can also be delays with network communication, so it is usually a good idea to invoke these web service requests in separate threads

Part II

GDI+
<table>
<thead>
<tr>
<th><strong>GDI+</strong></th>
<th><strong>Graphics</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- GDI+ (Graphics Device Interface) is a set of Windows libraries for drawing 2D graphics.</td>
<td></td>
</tr>
<tr>
<td>- The System.Drawing namespace provides managed access to these libraries.</td>
<td></td>
</tr>
<tr>
<td>- The Graphics class allows drawing of rectangles, ellipses, and other basic 2D drawing functions.</td>
<td>[demo]</td>
</tr>
</tbody>
</table>