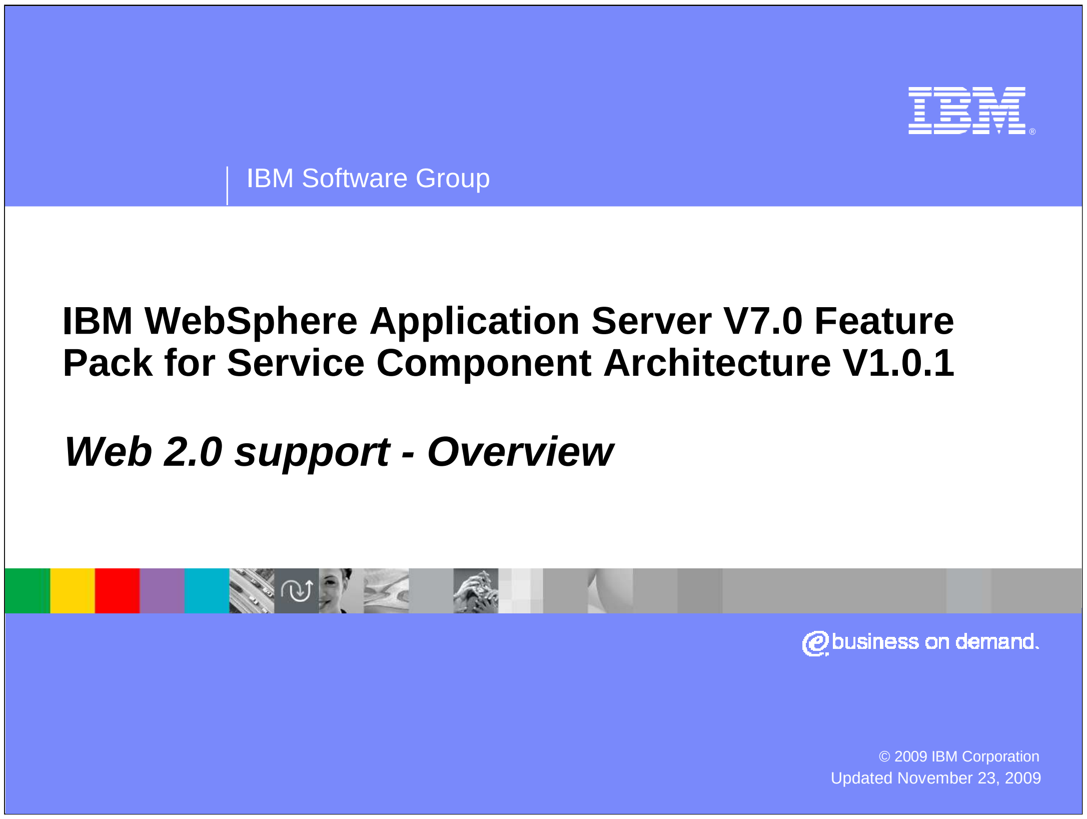
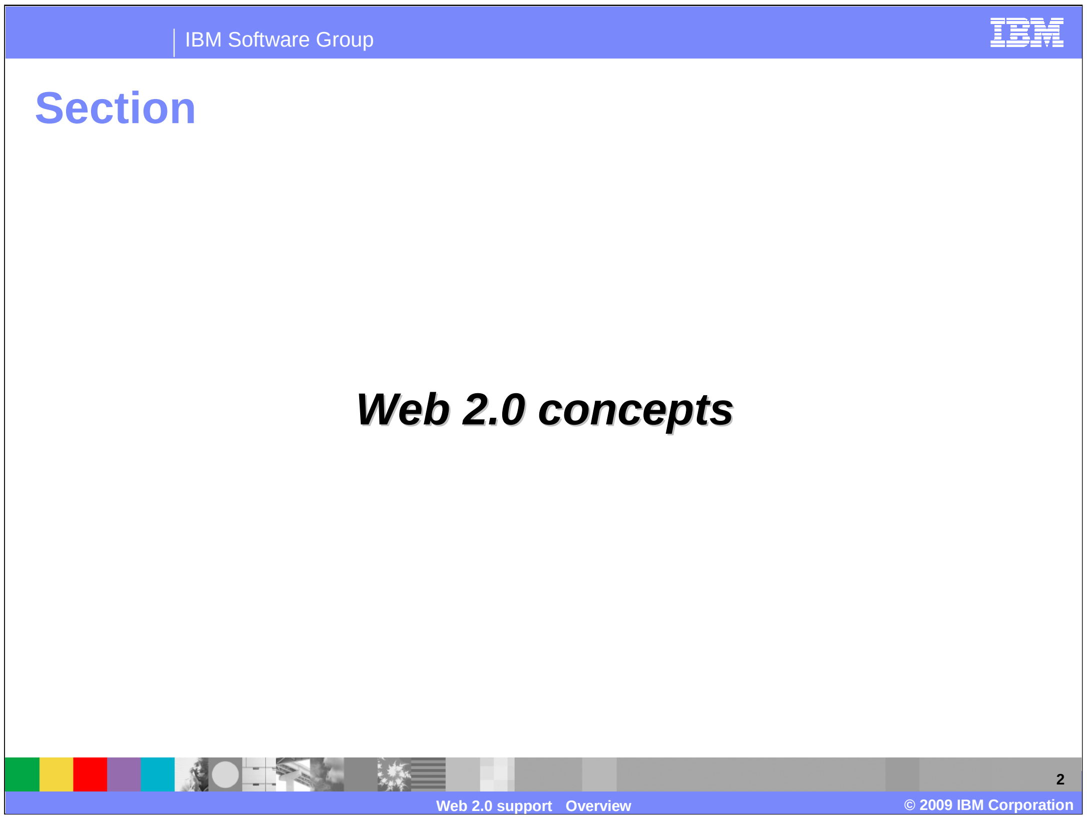


IBM Software Group

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture V1.0.1

Web 2.0 support - Overview

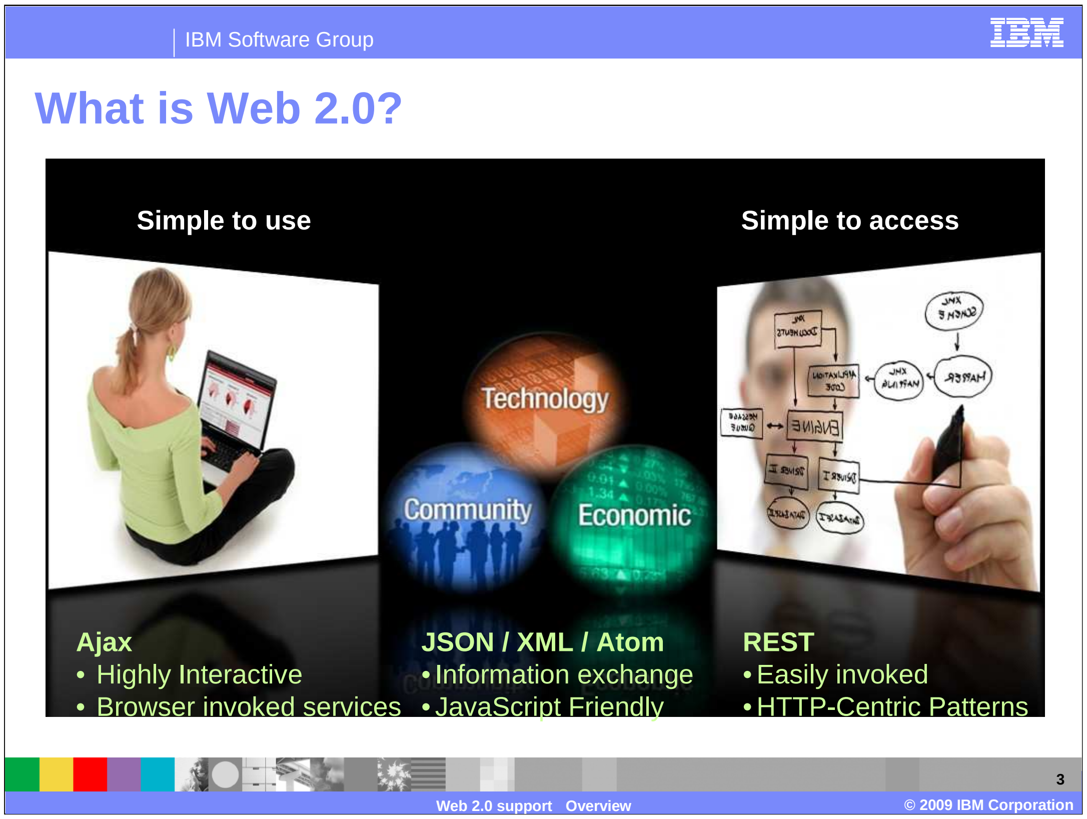
This presentation will talk about the overview of the Web 2.0 features that are part of the IBM WebSphere® Application Server Feature Pack for Service Component Architecture (SCA).



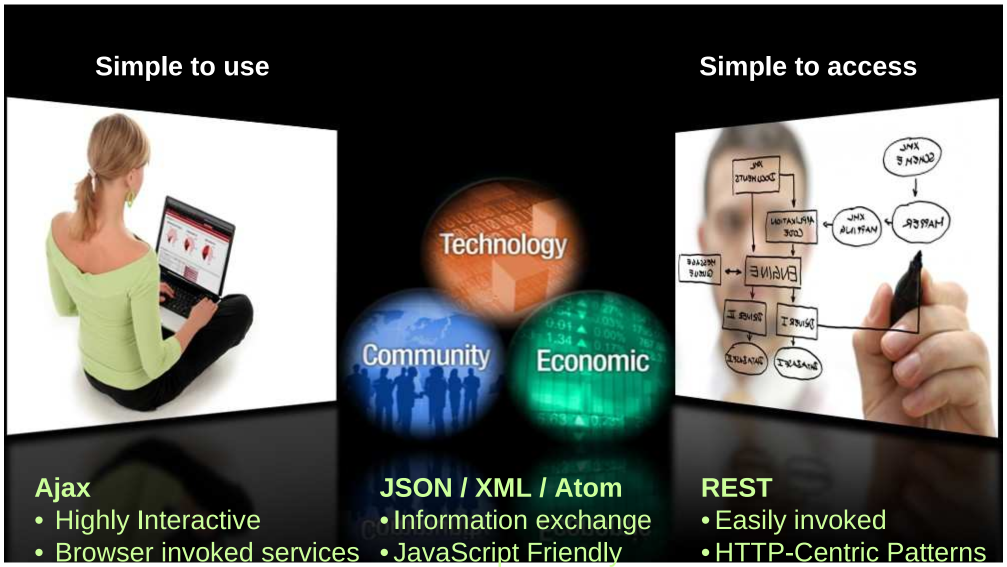
Section

Web 2.0 concepts

This presentation will start with a few slides on the concepts of Web 2.0. Note that you can reference the appendix for more Web 2.0 definitions



What is Web 2.0?



Ajax
- Highly Interactive
- Browser invoked services

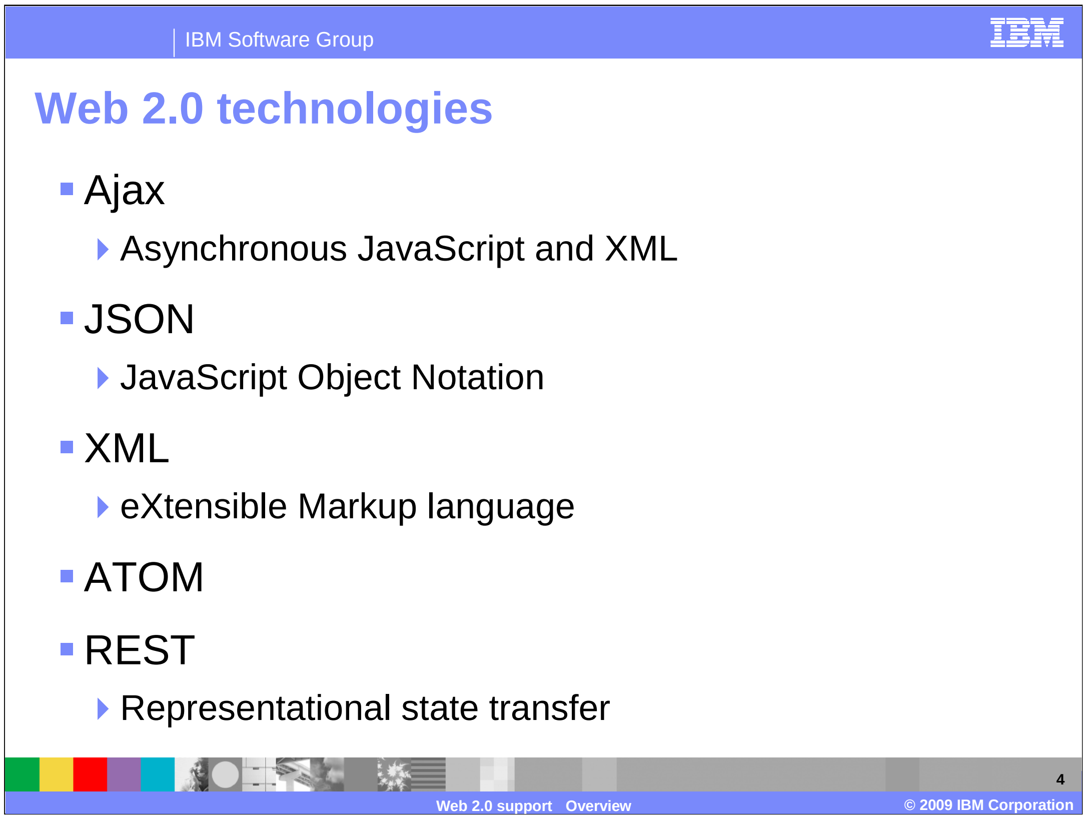
JSON / XML / Atom
- Information exchange
- JavaScript Friendly

REST
- Easily invoked
- HTTP-Centric Patterns

First, what is Web 2.0?

You are in an interesting time right now in development for the Web. For the first time in several years you have a new set of patterns that are letting you look at Web development in a new and different way. Rather than viewing the Web as a net of entire pages, where web-based applications generate a unique page at a time, you can now take a fundamentally different view of the web. You can now view the Web as a set of services that can be asynchronously invoked, merged together with other services, and rewoven into new, dynamic combinations. In addition, you are seeing an abundance of community software such as blogs, wikis, spaces, forums, and many others that are shaping the way users participate in the development life cycle. Agile development using techniques learned from the open source is also influencing the way software is built. As such an umbrella of patterns, technologies, and approaches as a whole has been coined Web 2.0, and it's a significantly different approach to the web.

Web 2.0 is a trend in the use of World Wide Web technology and Web design that aims to facilitate creativity, information sharing, and, most notably, collaboration among users. These concepts have led to the development and evolution of web-based communities and hosted services, such as social-networking sites, wikis, and blogs. If Web 1.0 was about connecting computers, Web 2.0 is about connecting people and encouraging collaboration in ways not possible before.


IBM Software Group

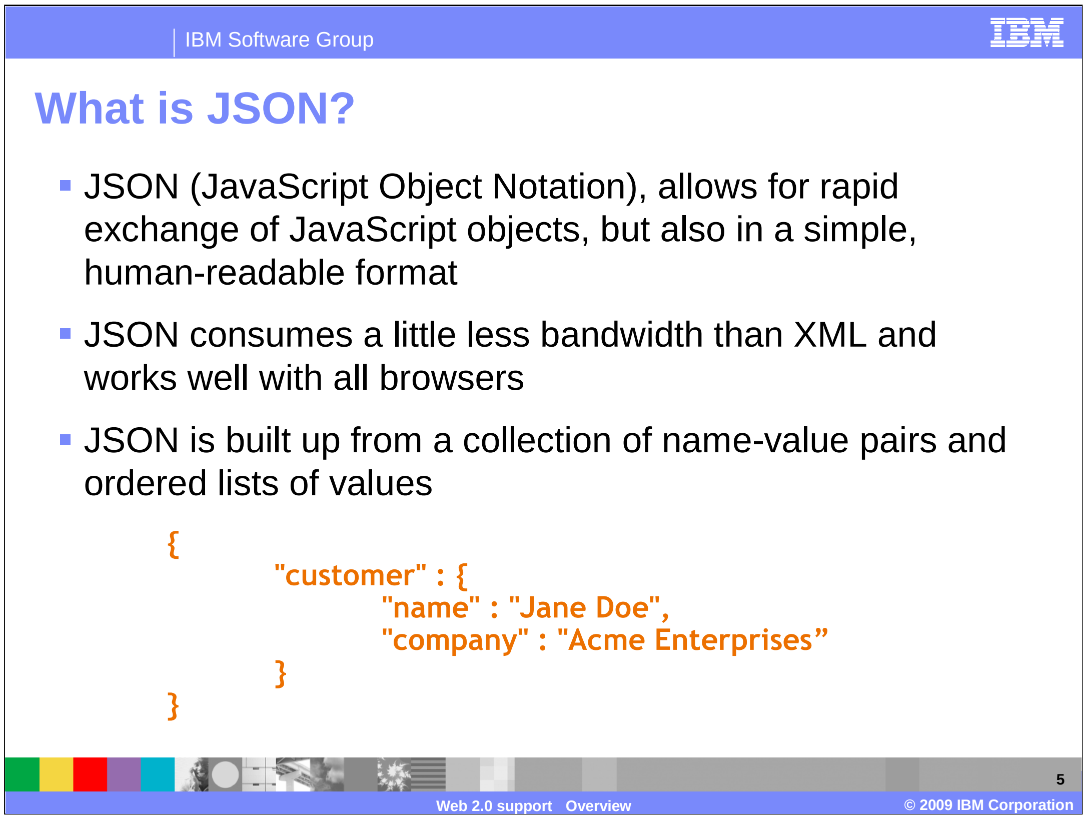
Web 2.0 technologies

- Ajax
  - Asynchronous JavaScript and XML
- JSON
  - JavaScript Object Notation
- XML
  - eXtensible Markup language
- ATOM
- REST
  - Representational state transfer

4

Web 2.0 support Overview © 2009 IBM Corporation

Web 2.0 technologies include Asynchronous JavaScript and XML (Ajax), JavaScript Object Notation (JSON), eXtensible Markup language (XML), Atom, and Representational state transfer (REST). The primary instigator of this new way of looking at the Web is a set of technologies commonly referred to as Ajax (Asynchronous JavaScript and XML). It's a way of building Web pages that, rather than mixing content and presentation together, separates the two. It separates the two into a basic page that contains the presentation (lists, trees, and so on.). In addition, a set of XML or JSON that are asynchronously fetched using a JavaScript Object called the XML HTTP request object. However, although X in Ajax represents XML, fetching XML is not required. Since Ajax architectures started using XML, it kept the name, but any content, including HTML, JavaScript, XML, or anything TEXT can be fetched asynchronously. It can also be used to populate the presentation based upon the actions of your of the page.


IBM Software Group

What is JSON?

- JSON (JavaScript Object Notation), allows for rapid exchange of JavaScript objects, but also in a simple, human-readable format
- JSON consumes a little less bandwidth than XML and works well with all browsers
- JSON is built up from a collection of name-value pairs and ordered lists of values

```
{
        "customer" : {
                "name" : "Jane Doe",
                "company" : "Acme Enterprises”
        }
}
```

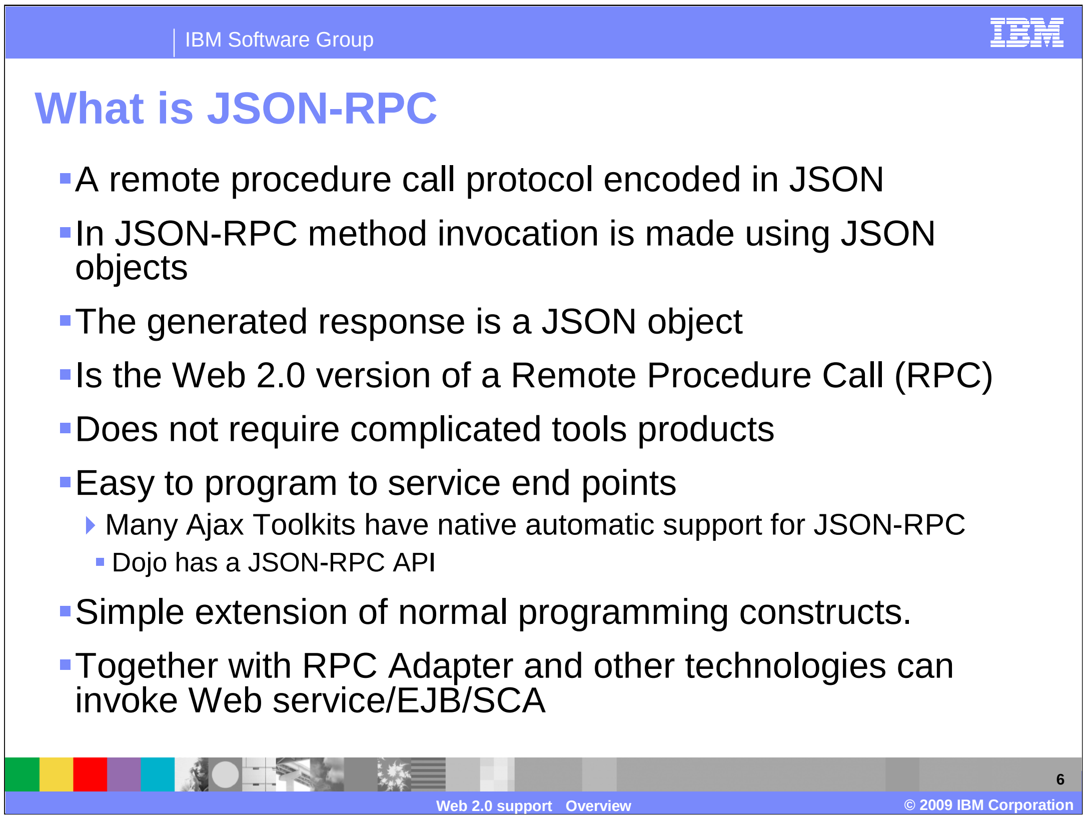
5

Web 2.0 support   Overview

© 2009 IBM Corporation

JSON is a lightweight computer data interchange format. It is a text-based, human-readable format for representing simple data structures and associative arrays (called objects). JSON Consumes a little less bandwidth than XML and works well with all browsers. JSON is built up from a collection of name-value pairs and ordered lists of values.

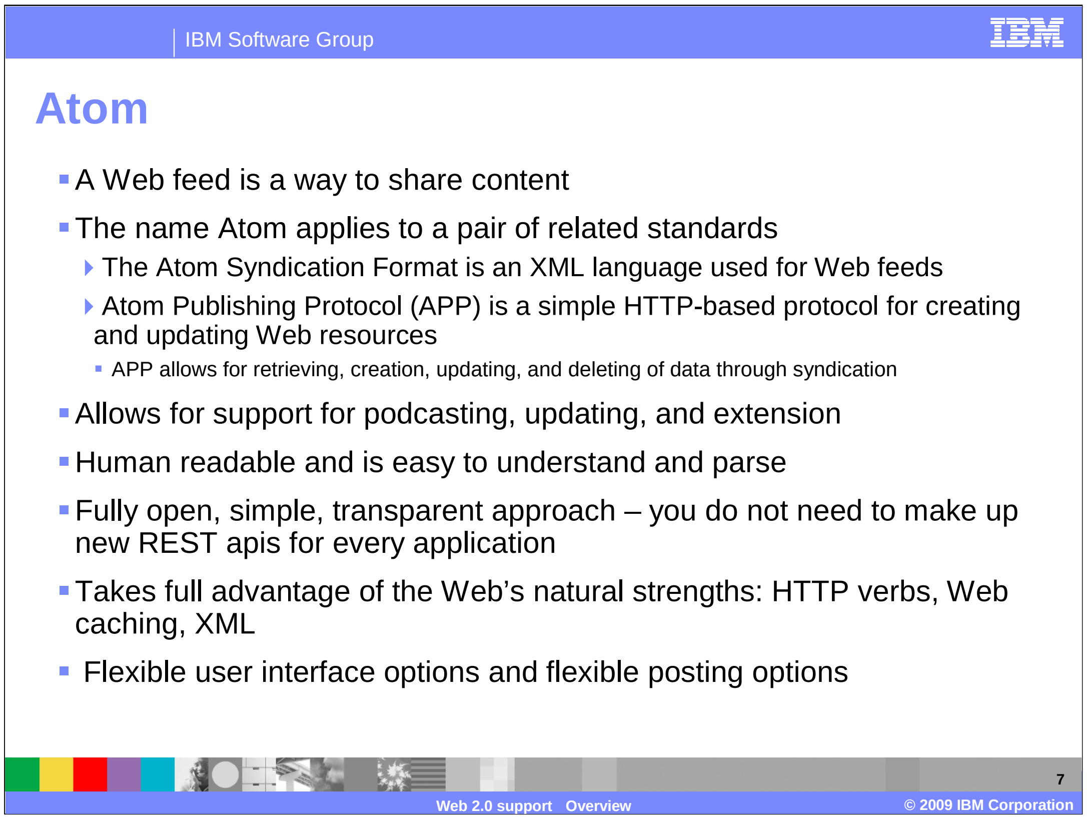
XML can be cumbersome in JavaScript to navigate so you move towards using JSON as a structured format. Unlike XML, which a browser has to parse as part of the DOM (Document Object Model), A JSON object becomes part of your code using eval. The JSON format is often used for transmitting structured data over a network connection in a process called serialization. Its main application is in Ajax Web application programming, where it serves as an alternative to the traditional use of the XML format. JSON is a simple, common representation of data that can be used for communication between servers and browser clients, communication between peers, and language independent data interchange.



What is JSON-RPC

- A remote procedure call protocol encoded in JSON
- In JSON-RPC method invocation is made using JSON objects
- The generated response is a JSON object
- Is the Web 2.0 version of a Remote Procedure Call (RPC)
- Does not require complicated tools products
- Easy to program to service end points
  - Many Ajax Toolkits have native automatic support for JSON-RPC
    - Dojo has a JSON-RPC API
- Simple extension of normal programming constructs.
- Together with RPC Adapter and other technologies can invoke Web service/EJB/SCA

Remote procedure call (RPC) is a technology that allows a computer program to cause a subroutine or procedure to start in another address space. The address space is commonly on another computer on a shared network without the programmer explicitly coding the details for this remote interaction. JSON-RPC is a remote procedure call protocol encoded in JSON. It is a very simple protocol (and very similar to XML-RPC), defining only a handful of data types and commands. In contrast to XML-RPC or SOAP, it allows for bidirectional communication between the service and the client, treating each more like peers and allowing peers to call one another or send notifications to one another. It also allows multiple calls to be sent to a peer which can be answered out of order. In JSON-RPC method, invocation is made using JSON objects. The generated response is a JSON object and the registered Java™ Bean can be accessed through Dojo's JSON-RPC API. JSON-RPC does not require complicated tools products, is easy to program to service end points, is a simple extension of normal programming constructs and can Invoke Web service/EJB/SCA.


IBM Software Group

Atom

- A Web feed is a way to share content
- The name Atom applies to a pair of related standards
  - The Atom Syndication Format is an XML language used for Web feeds
  - Atom Publishing Protocol (APP) is a simple HTTP-based protocol for creating and updating Web resources
    - APP allows for retrieving, creation, updating, and deleting of data through syndication
- Allows for support for podcasting, updating, and extension
- Human readable and is easy to understand and parse
- Fully open, simple, transparent approach – you do not need to make up new REST apis for every application
- Takes full advantage of the Web's natural strengths: HTTP verbs, Web caching, XML
- Flexible user interface options and flexible posting options

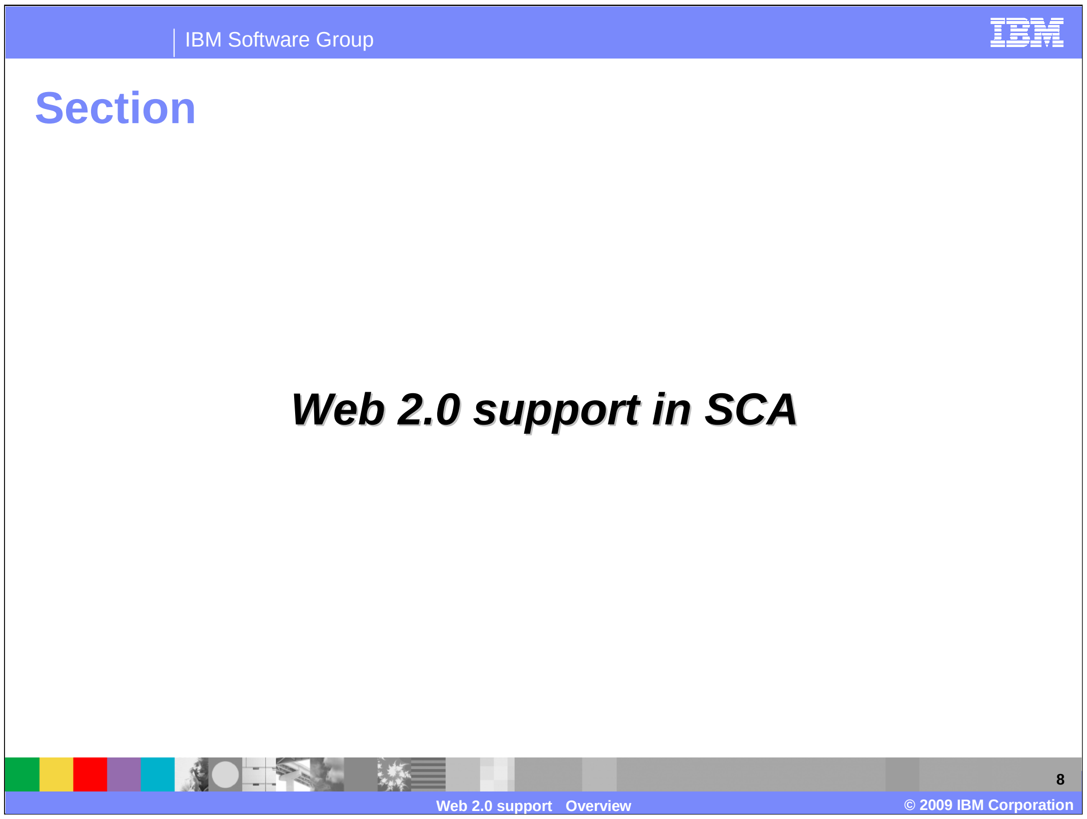
7

Web 2.0 support Overview © 2009 IBM Corporation

A Web **feed** is a data format used for providing users with frequently updated content. Content distributors *syndicate* a Web feed, thereby allowing users to *subscribe* to it. A Web feed is also sometimes referred to as a *syndicated feed*. Atom and RSS are two such Web feed formats. Web feed formats are used to publish frequently updated content such as blog entries, news headlines, and podcasts. The name **Atom** applies to a pair of related standards. The *Atom Syndication Format* is an XML language used for Web feeds, while the *Atom Publishing Protocol* (short *AtomPub* or *APP*) is a simple HTTP-based protocol for creating and updating Web resources. A feed contains entries, which can be headlines, full-text articles, excerpts, summaries, or links to content on a Web site, along with various metadata.

The Atom format was developed as an alternative to RSS. Atom Takes full advantage of the web's natural strengths: HTTP verbs, Web caching, and XML. Atom is supported by: Bloglines, Google (Blogger, News), Typepad, LiveJournal, NewsGator, FeedDemon, O'Reilly Developer Weblogs, Drupal, Flickr, Apple, Microsoft® and of course IBM. Atom allows for better support for podcasting, updating, and extension than RSS provided. Atom is also human readable and is easy to understand and parse. In addition, Atom, through the use of the Atom Publishing Protocol (APP) pushes syndication to the next level as you can perform updates to syndication.

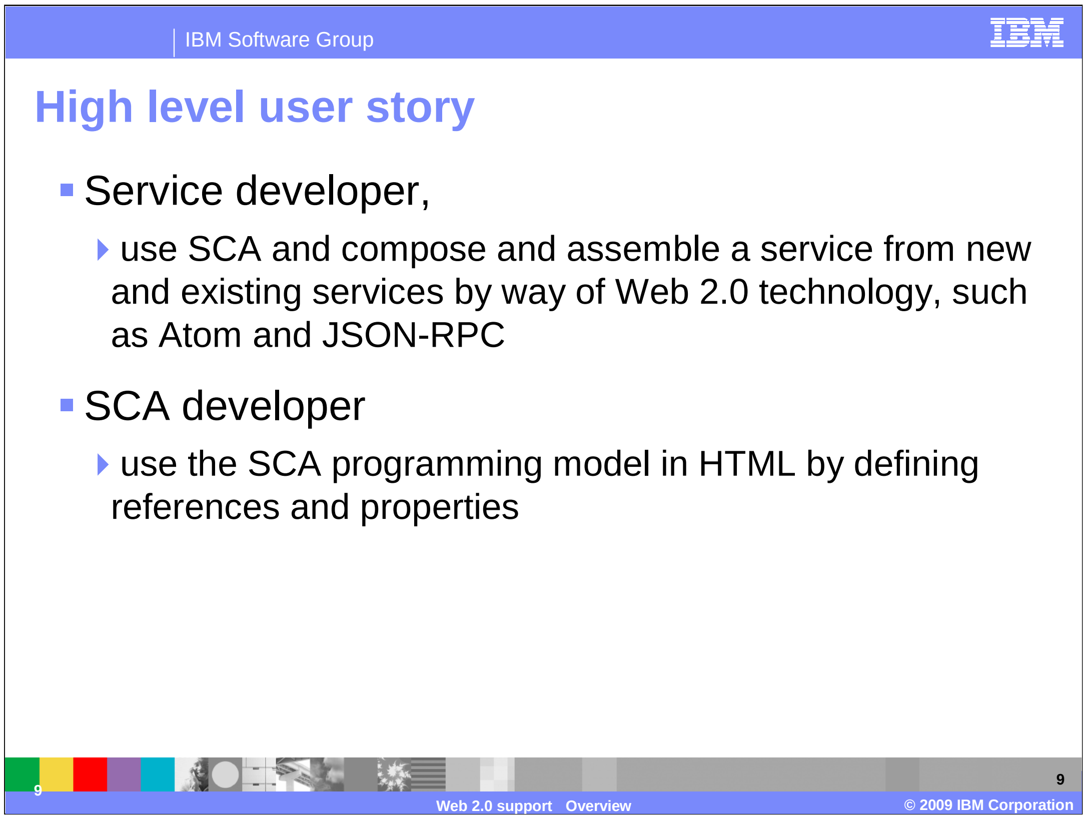
Contrary to popular use-cases, a feed is not always "the 15 most recent news articles". Atom can represent other resources and collections. You can include additional namespaces in an Atom feed. This means that you can represent all kinds of things such as Orders, Prospects, Documents, Blog posts, Calendar entries, Mail messages, Vacation schedules, Conference registrations, Expense approvals, and Activities.



Section

Web 2.0 support in SCA

The next section will cover the overview of Web 2.0 support in SCA.



IBM Software Group

High level user story

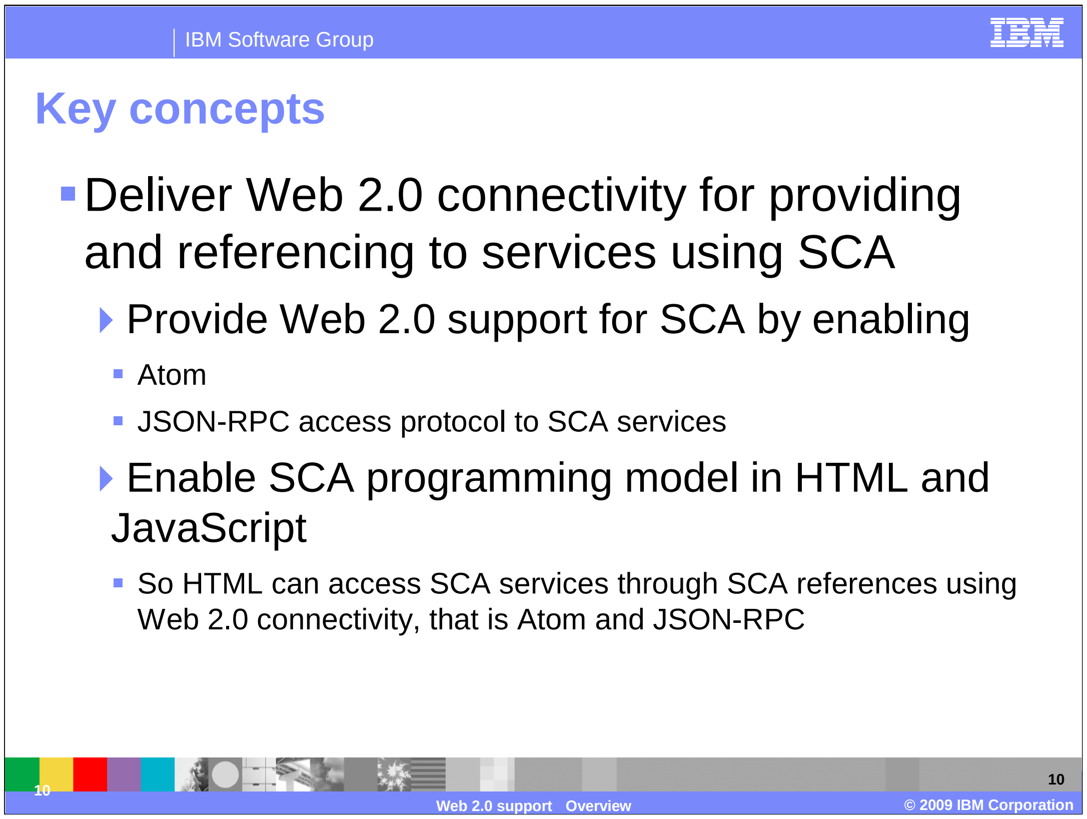
- Service developer,
  - use SCA and compose and assemble a service from new and existing services by way of Web 2.0 technology, such as Atom and JSON-RPC
- SCA developer
  - use the SCA programming model in HTML by defining references and properties

9

Web 2.0 support Overview © 2009 IBM Corporation

The Feature Pack for SCA covers three main high level user stories.

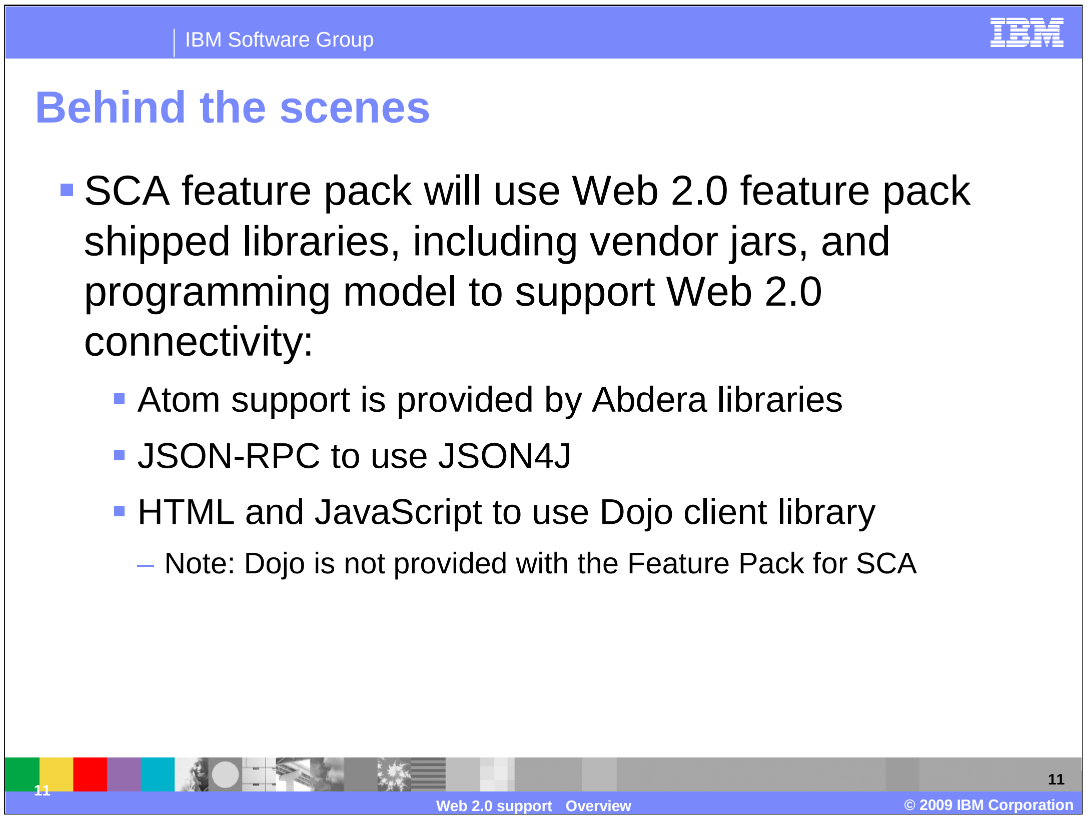
As a Service Developer, you want to use SCA and compose and assemble a service from new and existing services which are available by way of Web 2.0 technology, such as Atom and JSON-RPC. You want to make the newly assembled service available over Web 2.0 technology. This is so that you can create an open, implementation neutral, service oriented description of the newly created service assembly and composition. You can create this such that services that you provide or refer to reside on Web 2.0 technology. As an SCA developer you want to use the SCA programming model in HTML by defining references and properties.



Key concepts

- Deliver Web 2.0 connectivity for providing and referencing to services using SCA
  - Provide Web 2.0 support for SCA by enabling
    - Atom
    - JSON-RPC access protocol to SCA services
  - Enable SCA programming model in HTML and JavaScript
    - So HTML can access SCA services through SCA references using Web 2.0 connectivity, that is Atom and JSON-RPC

The Feature Pack for SCA will provide Web 2.0 support for SCA by enabling Atom and JSON-RPC access protocol to SCA services. It will also enable the SCA programming model in HTML and JavaScript so that that HTML can access SCA services through SCA references using Web 2.0 connectivity such as Atom and JSON-RPC.



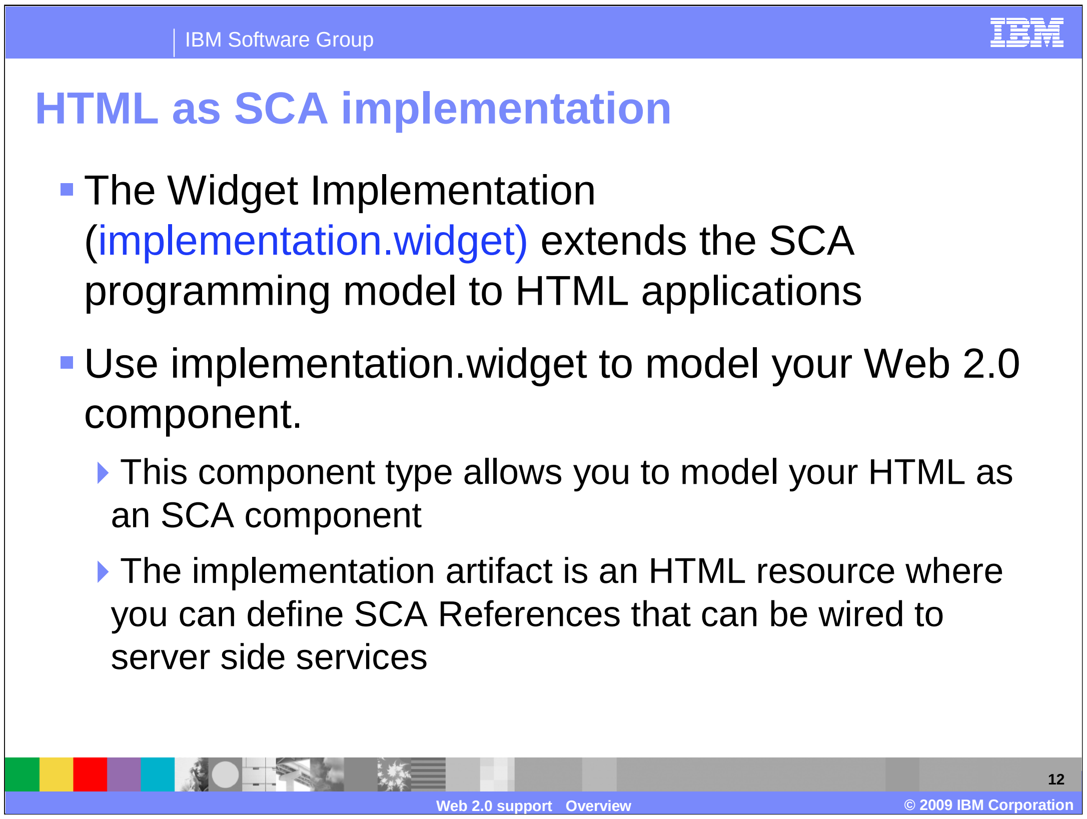
IBM Software Group

Behind the scenes

- SCA feature pack will use Web 2.0 feature pack shipped libraries, including vendor jars, and programming model to support Web 2.0 connectivity:
  - Atom support is provided by Abdera libraries
  - JSON-RPC to use JSON4J
  - HTML and JavaScript to use Dojo client library
    - Note: Dojo is not provided with the Feature Pack for SCA

11 11 Web 2.0 support Overview © 2009 IBM Corporation

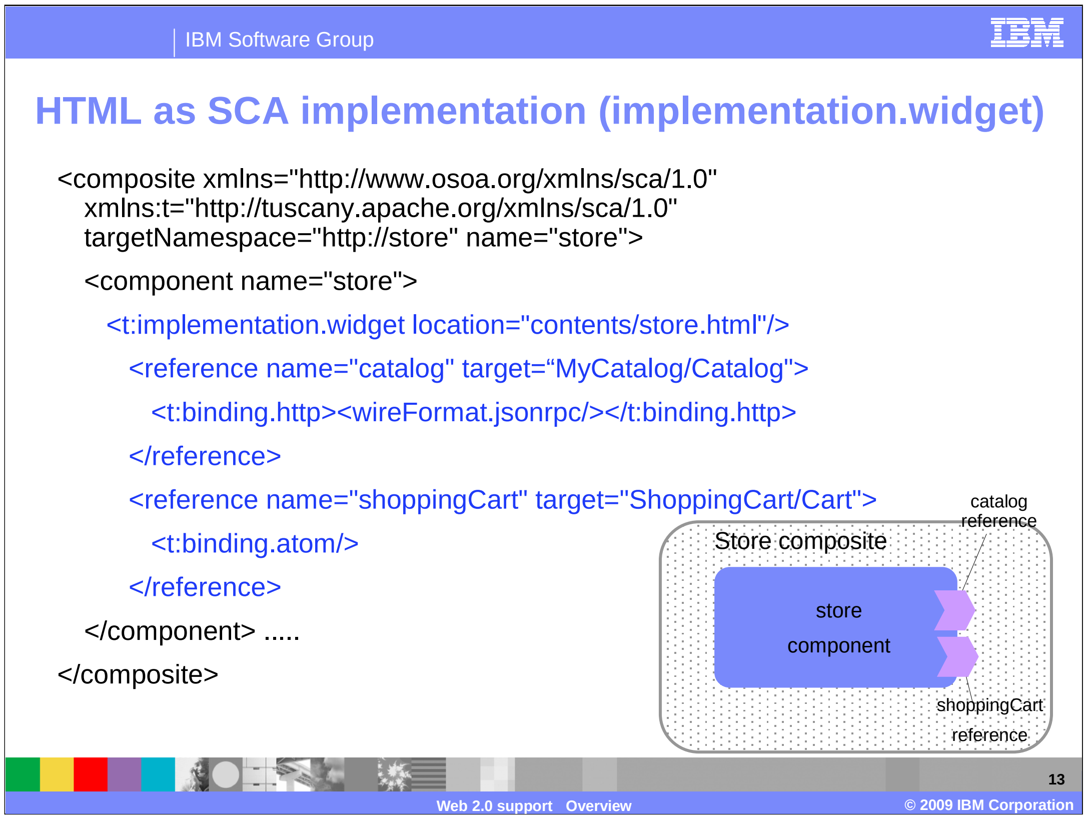
The Feature Pack for SCA will use the Feature Pack for Web 2.0 shipped libraries such as JSON4J and the blue-washed version of Dojo. It will also include vendor jars, and a programming model to support Web 2.0 connectivity. Specifically Atom support is provided by Abdera libraries, JSON-RPC will use JSON4J, and HTML and JavaScript will use the Dojo client library. Dojo is not provided with the Feature Pack for SCA, but you can easily get the blue-washed version of Dojo through the Feature Pack for Web 2.0.



HTML as SCA implementation

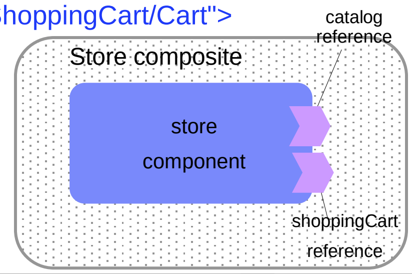
- The Widget Implementation (implementation.widget) extends the SCA programming model to HTML applications
- Use implementation.widget to model your Web 2.0 component.
  - This component type allows you to model your HTML as an SCA component
  - The implementation artifact is an HTML resource where you can define SCA References that can be wired to server side services

The Feature Pack for SCA includes support for the Widget Implementation, implementation.widget, which extends the SCA programming model to HTML applications. implementation.widget allows you to model your HTML as an SCA component. The implementation artifact is an HTML resource where you can define SCA References that can be wired to server side services. Currently, you can define references to remote services using and Atom binding and HTTP binding with wire format JSON-RPC.

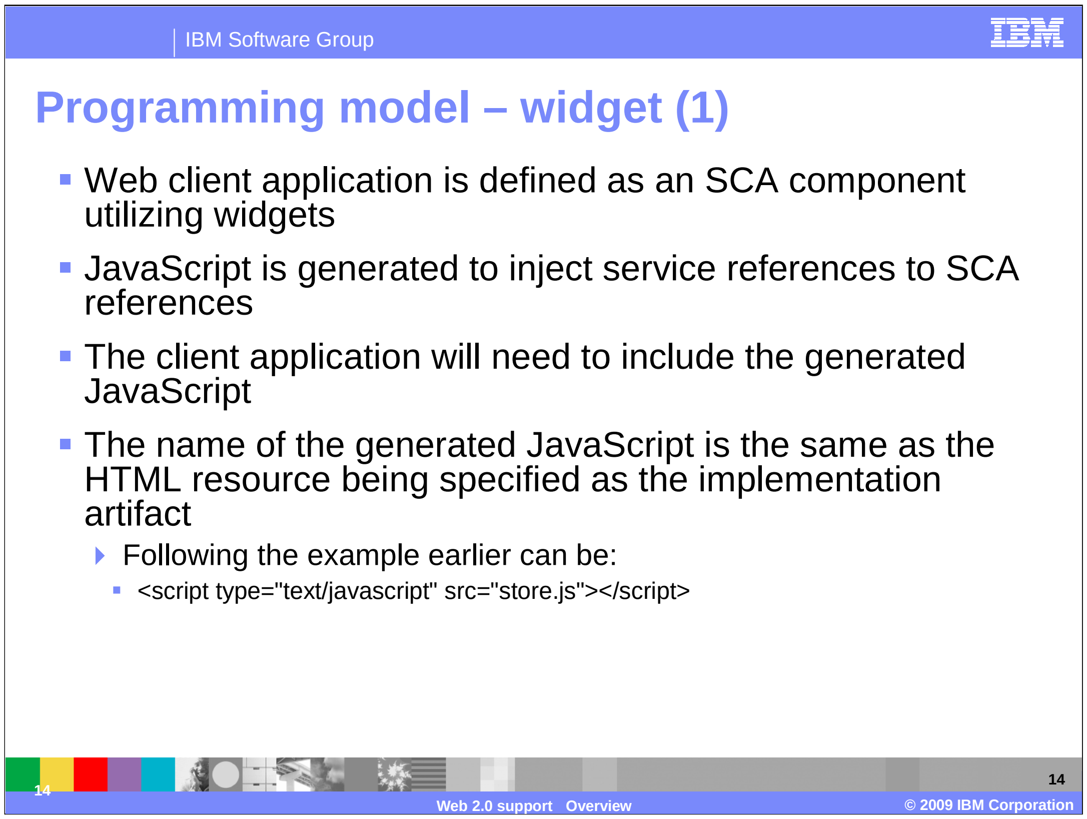


HTML as SCA implementation (implementation.widget)

```
<composite xmlns="http://www.osoa.org/xmlns/sca/1.0"
  xmlns:t="http://tuscany.apache.org/xmlns/sca/1.0"
  targetNamespace="http://store" name="store">
  <component name="store">
    <t:implementation.widget location="contents/store.html"/>
      <reference name="catalog" target=“MyCatalog/Catalog">
        <t:binding.http><wireFormat.jsonrpc/></t:binding.http>
      </reference>
      <reference name="shoppingCart" target="ShoppingCart/Cart">
        <t:binding.atom/>
      </reference>
  </component> .....
</composite>
```



The code above shows a sample composite file. The component store contains the implementation.widget. The implementation.widget component type allows you to model your HTML as an SCA component. The implementation artifact is an HTML resource, and above you can see how you can then define SCA References that can be wired to server side services. There are two references for the store component. They are catalog which is an HTTP binding with a wire format of JSON RPC, and shoppingCart which is a Atom binding.



Programming model – widget (1)

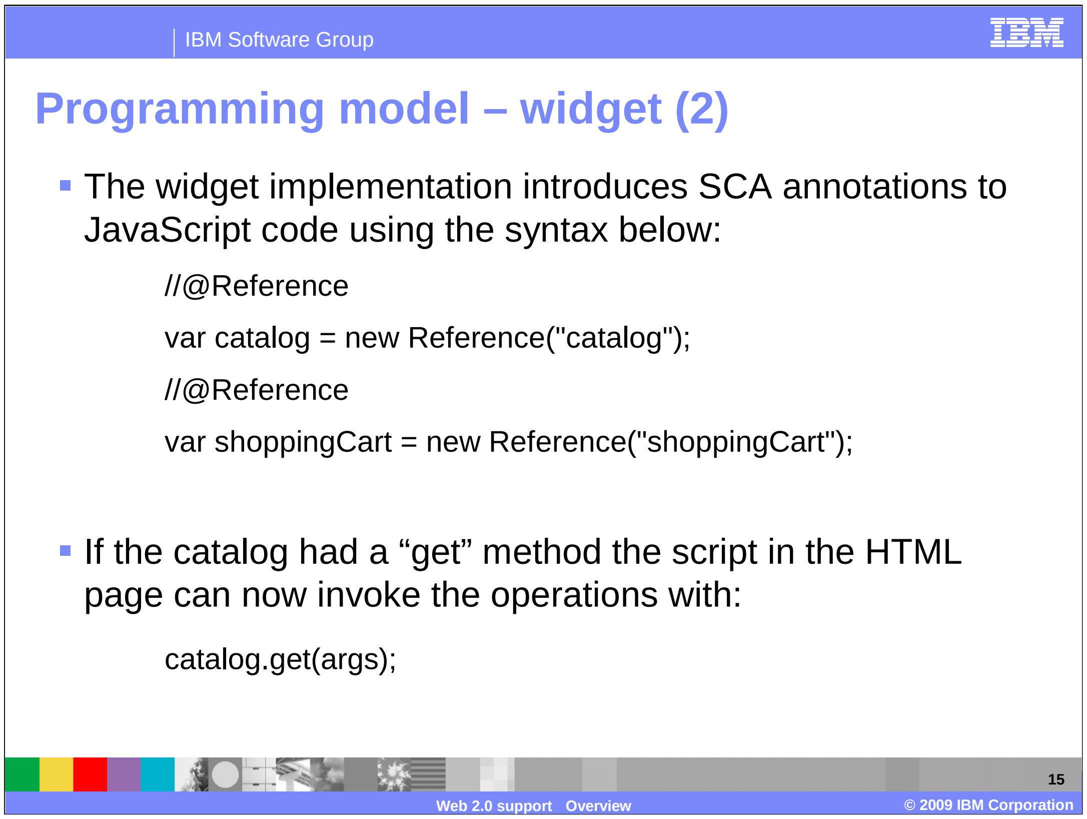
- Web client application is defined as an SCA component utilizing widgets
- JavaScript is generated to inject service references to SCA references
- The client application will need to include the generated JavaScript
- The name of the generated JavaScript is the same as the HTML resource being specified as the implementation artifact
  - Following the example earlier can be:
    - <script type="text/javascript" src="store.js"></script>

When your Web client application is defined as an SCA component utilizing widgets, a JavaScript file is generated which can be included within an HTML document. The JavaScript is to properly inject service references to SCA references defined on the same HTML document. The client application will need to include the generated JavaScript that will contain the necessary client proxy used to access the server side services. The name of the generated JavaScript is the same as the HTML resource being specified as the implementation artifact.

Following the example earlier where the component name was store you just add to your HTML resource the generated JavaScript file:

```
<script type="text/javascript" src="store.js"></script>
```

The JavaScript is transformed into Dojo Toolkit calls, that are binding specific. Including the generated JavaScript will initialize the proxys for the SCA services which can then be injected into SCA references to make requests to the server-side components.


IBM Software Group

Programming model – widget (2)

- The widget implementation introduces SCA annotations to JavaScript code using the syntax below:

```
//@Reference
var catalog = new Reference("catalog");
//@Reference
var shoppingCart = new Reference("shoppingCart");
```

- If the catalog had a “get” method the script in the HTML page can now invoke the operations with:

```
catalog.get(args);
```

15

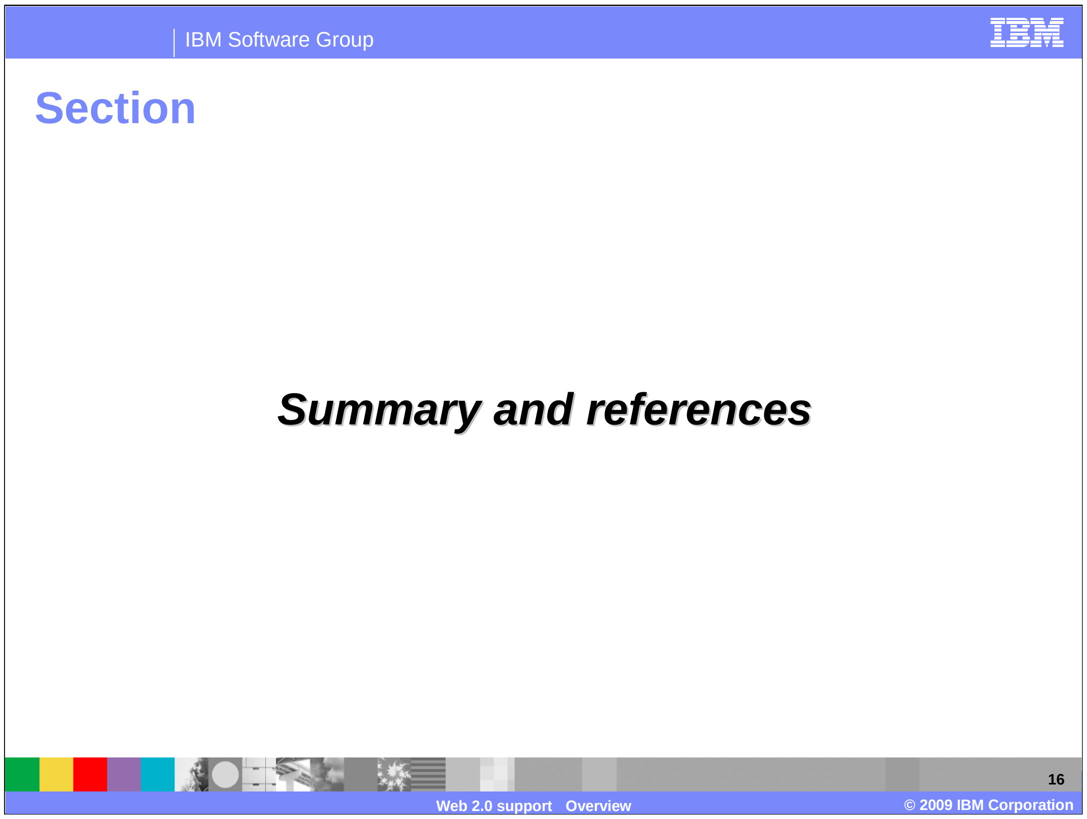
Web 2.0 support Overview © 2009 IBM Corporation

Using the widget implementation now introduces SCA annotations to the JavaScript code. Following the earlier example you can define the references

```
//@Reference
var catalog = new Reference("catalog");
//@Reference
var shoppingCart = new Reference("shoppingCart");
```

These references will get properly introspected by the implementation.widget and wired to the proper server side services.

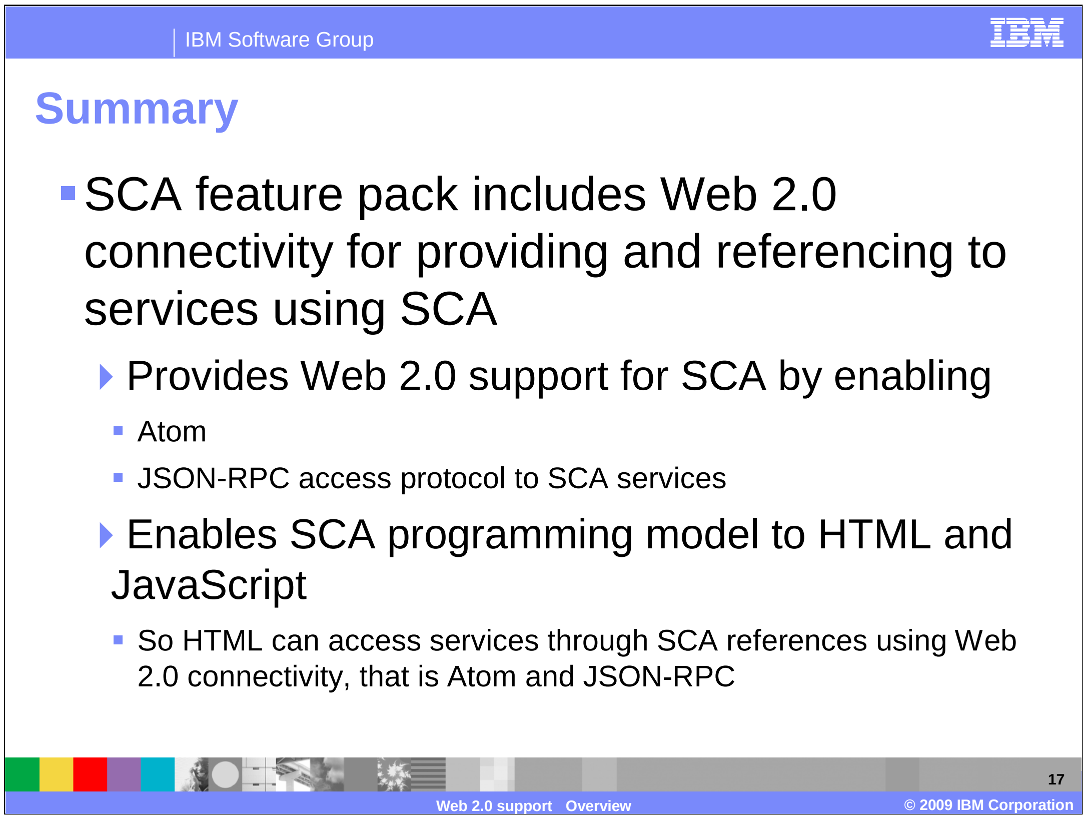
If the catalog had a get() method the script in the HTML page can now easily invoke the get method by calling catalog.get(args);



Section

Summary and references

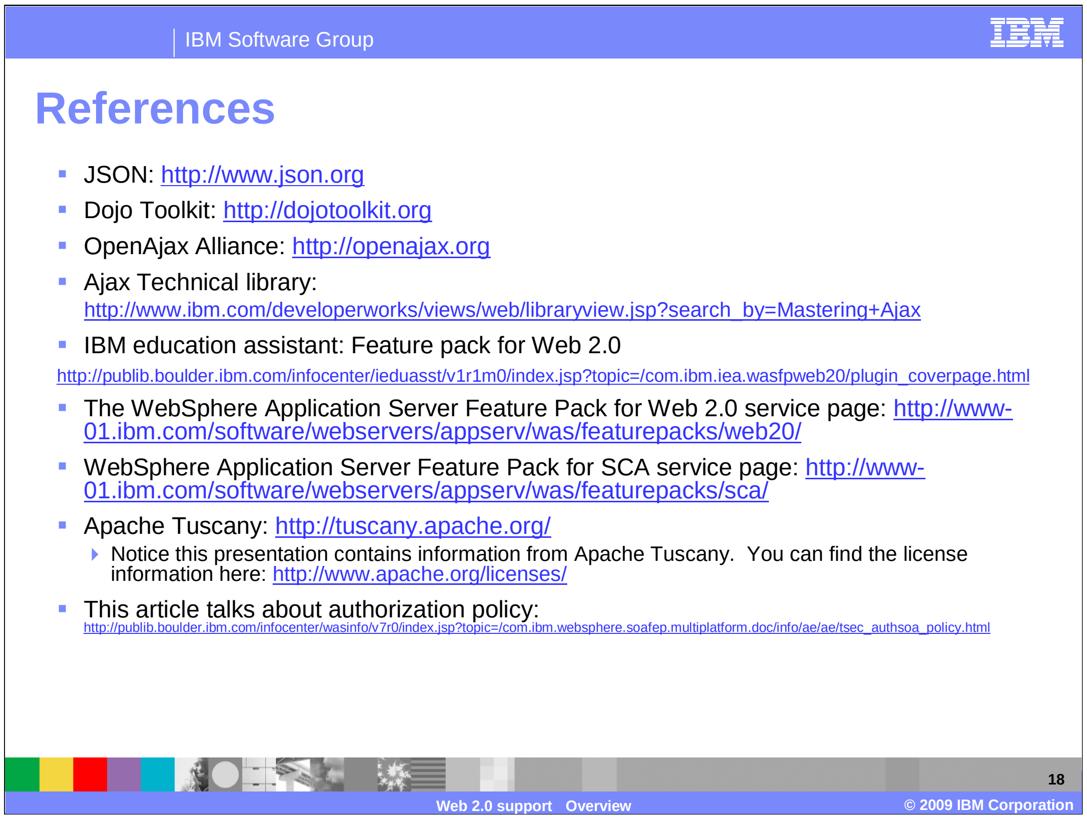
Next is summary of what you have learned and some references.



Summary

- SCA feature pack includes Web 2.0 connectivity for providing and referencing to services using SCA
  - Provides Web 2.0 support for SCA by enabling
    - Atom
    - JSON-RPC access protocol to SCA services
  - Enables SCA programming model to HTML and JavaScript
    - So HTML can access services through SCA references using Web 2.0 connectivity, that is Atom and JSON-RPC

The Feature Pack for SCA includes Web 2.0 connectivity for providing and referencing to services using SCA. It provides Web 2.0 support for SCA by enabling Atom and the JSON-RPC access protocol to SCA services. The feature pack enables SCA programming model to HTML and JavaScript so HTML can access services through SCA references using Web 2.0 connectivity, that is Atom and JSON-RPC.


IBM Software Group

References

- JSON: http://www.json.org
- Dojo Toolkit: http://dojotoolkit.org
- OpenAjax Alliance: http://openajax.org
- Ajax Technical library: http://www.ibm.com/developerworks/views/web/libraryview.jsp?search_by=Mastering+Ajax
- IBM education assistant: Feature pack for Web 2.0 http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.wasfpweb20/plugin_coverpage.html
- The WebSphere Application Server Feature Pack for Web 2.0 service page: http://www-01.ibm.com/software/webservers/appserv/was/featurepacks/web20/
- WebSphere Application Server Feature Pack for SCA service page: http://www-01.ibm.com/software/webservers/appserv/was/featurepacks/sca/
- Apache Tuscany: http://tuscany.apache.org/
  - Notice this presentation contains information from Apache Tuscany. You can find the license information here: http://www.apache.org/licenses/
- This article talks about authorization policy: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.soafep.multiplatform.doc/info/ae/ae/tsec_authsoa_policy.html

18

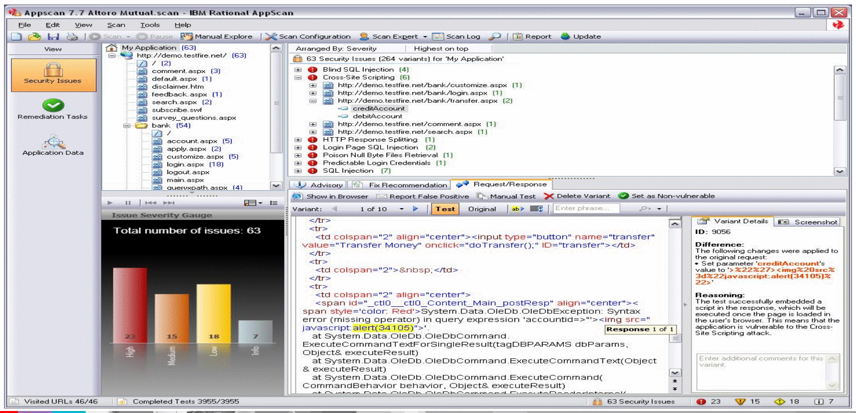
Web 2.0 support Overview © 2009 IBM Corporation

Above are some useful references.

Web 2.0 vulnerability testing

watchfire® an IBM company

- Explicit Security vulnerability testing is critical when working with JavaScript/Web2.0
- Rational® AppScan® (www.watchfire.com) is the industry-leading security vulnerability scanning tool for Ajax



Arranged By: Severity | Highest on top

63 Security Issues (264 variants) for 'My Application'

- Blind SQL Injection (4)
- Cross-Site Scripting (6)
  - http://demo.testfire.net/bank/customize.aspx (1)
  - http://demo.testfire.net/bank/login.aspx (1)
  - http://demo.testfire.net/bank/transfer.aspx (2)
    - creditAccount
    - debitAccount
  - http://demo.testfire.net/comment.aspx (1)
  - http://demo.testfire.net/search.aspx (1)
- HTTP Response Splitting (1)
- Login Page SQL Injection (2)
- Poison Null Byte Files Retrieval (1)
- Predictable Login Credentials (1)
- SQL Injection (7)

```
  </tr>
  <tr>
    <td colspan="2" align="center"><input type="button" name="transfer"
value="Transfer Money" onclick="doTransfer();" ID="transfer"></td>
  </tr>
  <tr>
    <td colspan="2">&nbsp;</td>
  </tr>
  <tr>
    <td colspan="2" align="center">
    <span id="_ctl0__ctl0_Content_Main_postResp" align="center"><
span style='color: Red'>System.Data.OleDb.OleDbException: Syntax
error (missing operator) in query expression 'accountid=>"'><img src="
javascript:alert(34105)">'.
   at System.Data.OleDb.OleDbCommand.
ExecuteCommandTextForSingleResult(tagDBPARAMS dbParams,
Object& executeResult)
   at System.Data.OleDb.OleDbCommand.ExecuteCommandText(Object
& executeResult)
   at System.Data.OleDb.OleDbCommand.ExecuteCommand(
CommandBehavior behavior, Object& executeResult)
```

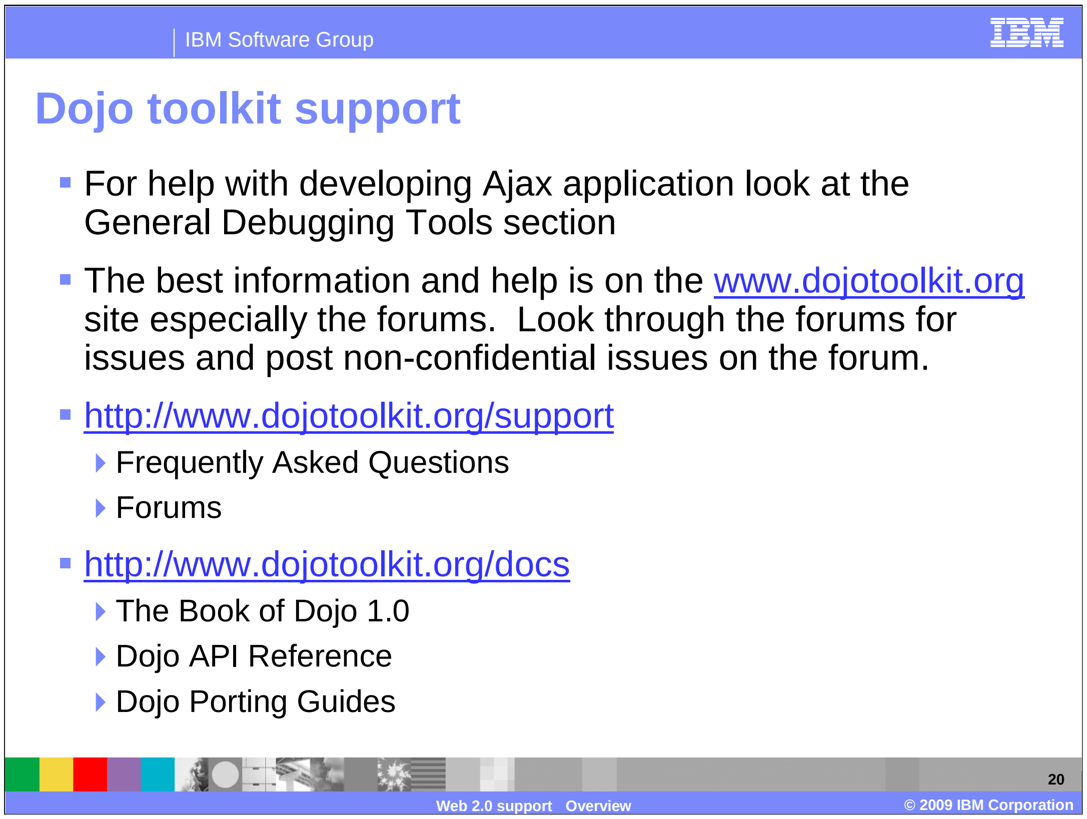
ID: 9056

Difference:
The following changes were applied to the original request:
• Set parameter 'creditAccount''s value to '>%22%27><img%20src%3d%22javascript:alert(34105)%22>'

Reasoning:
The test successfully embedded a script in the response, which will be executed once the page is loaded in the user's browser. This means that the application is vulnerable to the Cross-Site Scripting attack.

Enter additional comments for this variant.

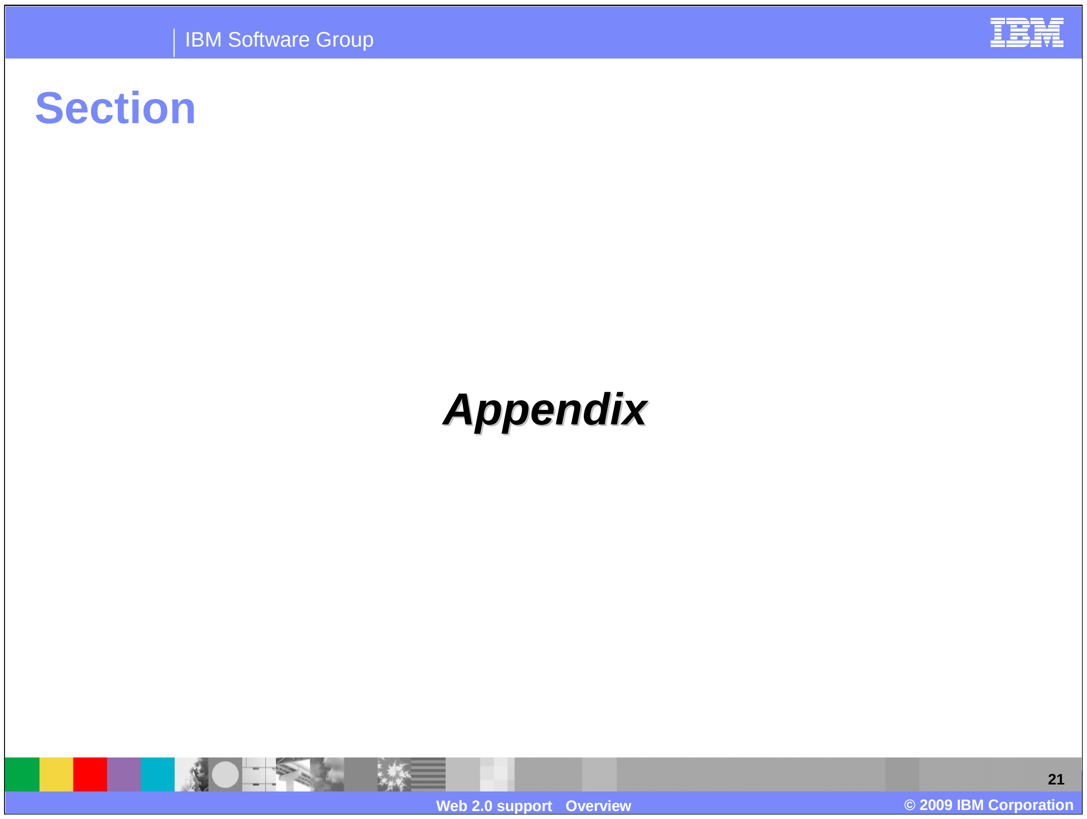
Outside-in security testing should be part of your quality assurance plan, watchfire.com is a security vulnerability testing tool.



Dojo toolkit support

- For help with developing Ajax application look at the General Debugging Tools section
- The best information and help is on the www.dojotoolkit.org site especially the forums. Look through the forums for issues and post non-confidential issues on the forum.
- http://www.dojotoolkit.org/support
  - Frequently Asked Questions
  - Forums
- http://www.dojotoolkit.org/docs
  - The Book of Dojo 1.0
  - Dojo API Reference
  - Dojo Porting Guides

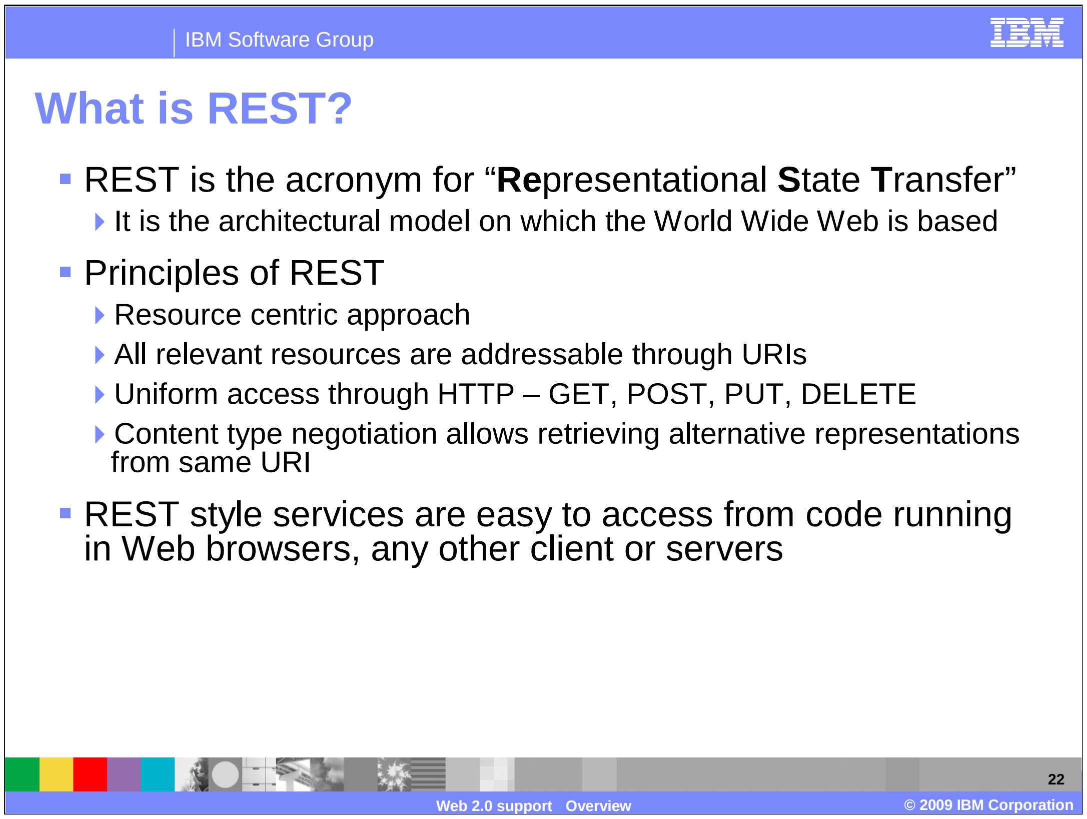
Here are some useful links for Dojo Toolkit.



Section

Appendix

Now that you have had a quick overview of Web 2.0 and Web 2.0 with SCA, the next couple of slides are more concepts of Web 2.0"



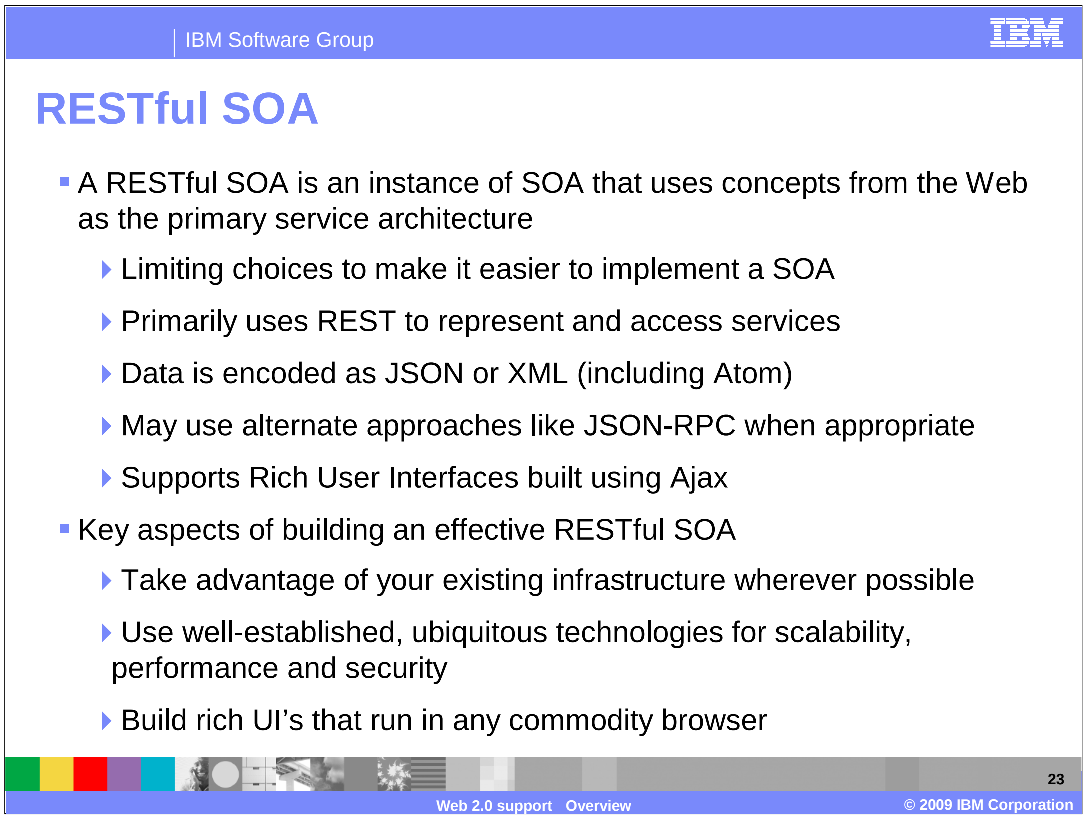
What is REST?

- REST is the acronym for "**Re**presentational **S**tate **T**ransfer"
  - It is the architectural model on which the World Wide Web is based
- Principles of REST
  - Resource centric approach
  - All relevant resources are addressable through URIs
  - Uniform access through HTTP – GET, POST, PUT, DELETE
  - Content type negotiation allows retrieving alternative representations from same URI
- REST style services are easy to access from code running in Web browsers, any other client or servers

Representational state transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web. The terms "representational state transfer" and "REST" were introduced in 2000 in the doctoral dissertation of Roy Fielding, one of the principal authors of the Hypertext Transfer Protocol (HTTP) specification. The terms have since come into widespread use in the networking community. REST strictly refers to a collection of network architecture principles which outline how resources are defined and addressed. The term is often used in a looser sense to describe any simple interface which transmits domain-specific data over HTTP without an additional messaging layer such as SOAP or session tracking by way of HTTP cookies. Systems which follow Fielding's REST principles are often referred to as "RESTful".

REST is a resource centric approach and all relevant resources are addressable by way of URIs. Uniform access by way of HTTP are through simple GET, POST, PUT, DELETE commands. GET returns a state representation of the identified resource. POST performs some form of application-specific update to the identified resource. PUT creates a new resource at an identified location (URI), and DELETE destroys a resource at the identified location (URI). Content type negotiation allows retrieving alternative representations from same URI. REST style services are easy to access from code running in Web browsers, any other client or servers. REST is very popular in the context of Ajax and can take full advantage of the World Wide Web caching infrastructure. REST can serve multiple representations of the same resource.

REST is the first key component in an implementation of a Web Oriented Architecture. REST is a style of architecture that is best exemplified within the HTTP protocol and enables the creation of Web services that are simple to implement, built for re-use, and are ultra-scalable. The simplicity of REST comes from the use of the fixed protocol (HTTP), fixed encryption model (HTTPs), and fixed identity token exchange (Basic-Auth or standard HTTP schemes).



RESTful SOA

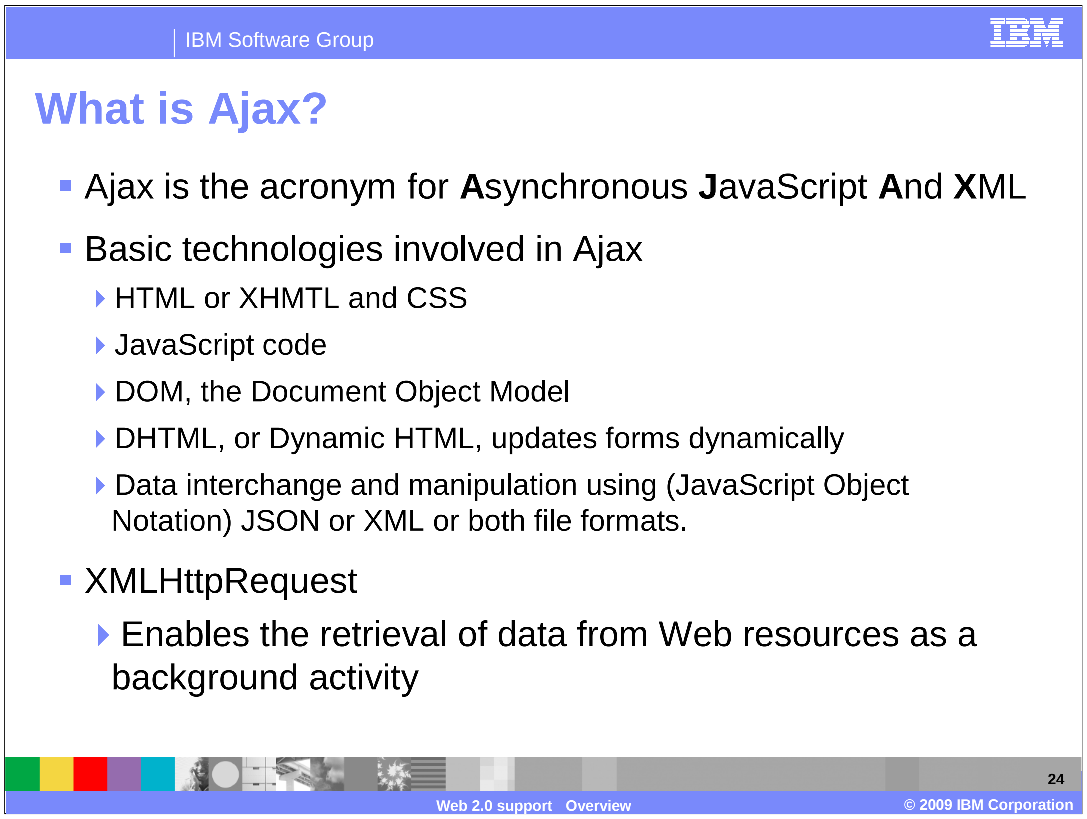
- A RESTful SOA is an instance of SOA that uses concepts from the Web as the primary service architecture
  - Limiting choices to make it easier to implement a SOA
  - Primarily uses REST to represent and access services
  - Data is encoded as JSON or XML (including Atom)
  - May use alternate approaches like JSON-RPC when appropriate
  - Supports Rich User Interfaces built using Ajax
- Key aspects of building an effective RESTful SOA
  - Take advantage of your existing infrastructure wherever possible
  - Use well-established, ubiquitous technologies for scalability, performance and security
  - Build rich UI's that run in any commodity browser

A RESTful Service-oriented Architecture SOA (sometimes referred to by the industry as WOA or Web Oriented Architecture) is just one implementation of a SOA where the Web is the SOA platform. IBM's position has always been that SOA is an architecture and you have several ways you can build an SOA, weather it be Web services, SCA, or EJB3 POJO's. RESTful SOA is another implementation of the architecture and extends the reach of your enterprise SOA putting services in the hands of the masses. The goal is to expose resources using HTTP, using a RESTful pattern, and Web formats that are easily consumed by the Web. You want to use your existing Web Infrastructure, which can be already hardened, to take advantage of the Internets scalability.

The primary instigator of this new way of looking at the Web is Ajax. This set of technologies has led you to have to take a step back and think about what this means for enterprise computing. You used to have to consider that enterprise services either ended at the corporate firewall, or at most were exposed under strict security to a limited set of corporate partners. You can now envision a time where some services extend all the way out to the browser. This is leading to a consideration of a new type of architecture, the Web-Oriented architecture. This architecture is an emerging term that was coined by Nick Gall that refers to a small set of services protocols that are optimized for browser and other end-client interactions. The way to look at this is to take the basic idea of services extended all the way out to the browser, and then consider the impact this has on the notion of an SOA. WOA refers to delivering simple XML Web services delivered over HTTP using the REST approach. REST is the foundation of a WOA. The next layer down is how you represent the information flowing from the service using technologies like Atom feeds and JSON. Finally, Ajax (that started the whole thing) is the way to render the information in the service to you. However, the core idea remains that you are only considering a small set of protocols aimed at bringing information out directly to the client.

Collectively, the technologies can be used to create a radically simplified service platform. Using REST and Atom to form the basis of the service invocation model. JSON and XML as the data interchange format. Ajax as the model for a rich client. This platform can also be used to create an architectural "bridge" between WOA and enterprise SOA, allowing services from the enterprise to be simplified and reach the masses by way of the web. You can refer to this as "extending your SOA to the Web".

SOA is an integral part of realizing the business benefit from Web 2.0. It enables businesses to unlock their enterprise content to reach new markets, to enable collective intelligence of communities, and take advantage of Web 2.0 technologies for collaboration.


IBM Software Group

What is Ajax?

- Ajax is the acronym for **A**synchronous **J**avaScript **A**nd **X**ML
- Basic technologies involved in Ajax
  - HTML or XHMTL and CSS
  - JavaScript code
  - DOM, the Document Object Model
  - DHTML, or Dynamic HTML, updates forms dynamically
  - Data interchange and manipulation using (JavaScript Object Notation) JSON or XML or both file formats.
- XMLHttpRequest
  - Enables the retrieval of data from Web resources as a background activity

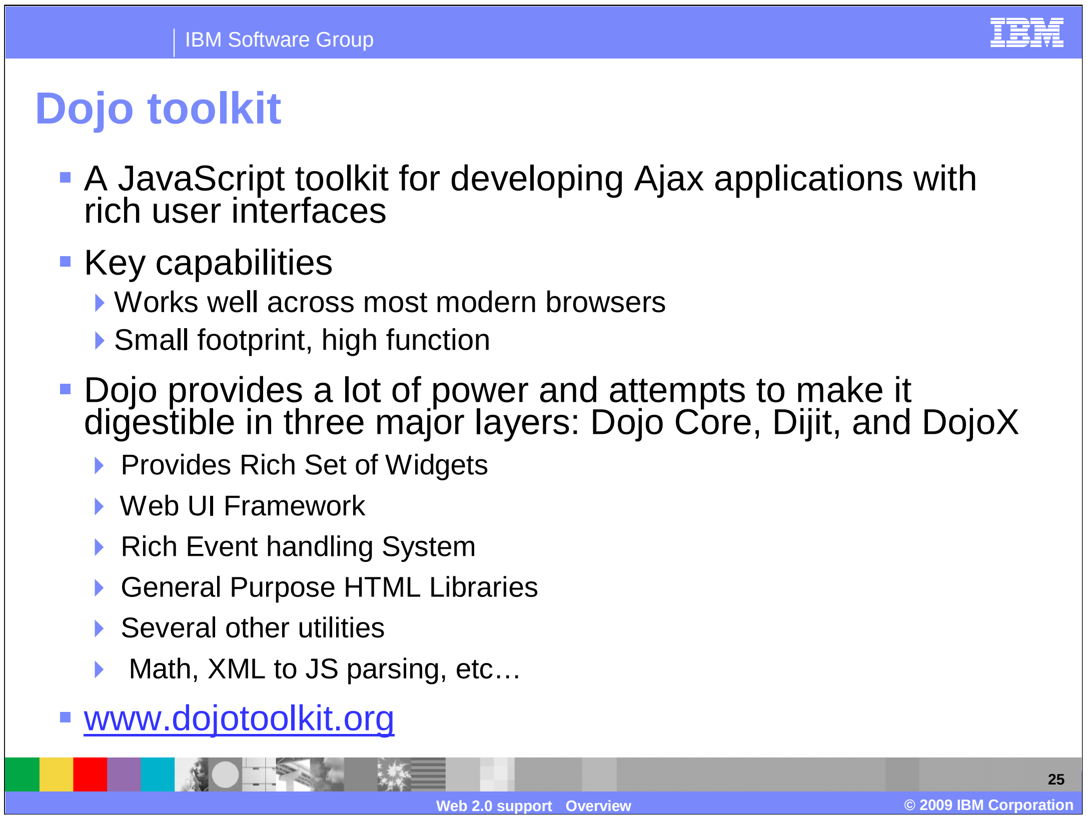
24

Web 2.0 support Overview © 2009 IBM Corporation

So what is Ajax? Ajax stands for Asynchronous JavaScript and XML. It is a group of inter-related Web development techniques for creating interactive Web applications. A primary characteristic is the increased responsiveness and interactivity of Web pages achieved by exchanging small amounts of data with the server. This is done "behind the scenes" so that entire Web pages do not have to be reloaded each time there is a need to fetch data from the server. This is intended to increase the Web page's interactivity, speed, functionality and usability.

Ajax is a cross-platform technique usable on many different operating systems, computer architectures, and Web browsers. It is a pattern for programming rich browser applications that uses open standards and can be mixed in with classic Web user interface. The Ajax technique uses a combination of elements. HTML or XHMTL is used to build Web forms and identify fields for use in an application. JavaScript code is the core code running Ajax applications and it helps facilitate communication with server applications. JavaScript is the scripting language in which Ajax function calls are typically made. DOM, the Document Object Model, is used (through JavaScript code) to work with both the structure of the HTML and (in some cases) XML returned from the server. DHTML, or Dynamic HTML, updates forms dynamically. Ajax accomplishes data interchange and manipulation using JSON or XML or both file formats. XML is sometimes used as the format for transferring data between the server and client; although, any format works, including preformatted HTML, plain text, and JSON.

Ajax is asynchronous, in that extra data is requested from the server and loaded in the background without interfering with the display and behavior of the existing page. The XMLHttpRequest object is used to exchange data asynchronously with the Web server. Ajax supports a rich client interaction model that is intuitive, responsive, and timely. The interaction is comparable to desktop applications. Continuous user interaction with event driven server processing and dynamic content refresh versus interrupted interaction with request driven server processing followed by static page refresh.



IBM Software Group

Dojo toolkit

- A JavaScript toolkit for developing Ajax applications with rich user interfaces
- Key capabilities
  - Works well across most modern browsers
  - Small footprint, high function
- Dojo provides a lot of power and attempts to make it digestible in three major layers: Dojo Core, Dijit, and DojoX
  - Provides Rich Set of Widgets
  - Web UI Framework
  - Rich Event handling System
  - General Purpose HTML Libraries
  - Several other utilities
  - Math, XML to JS parsing, etc…
- www.dojotoolkit.org

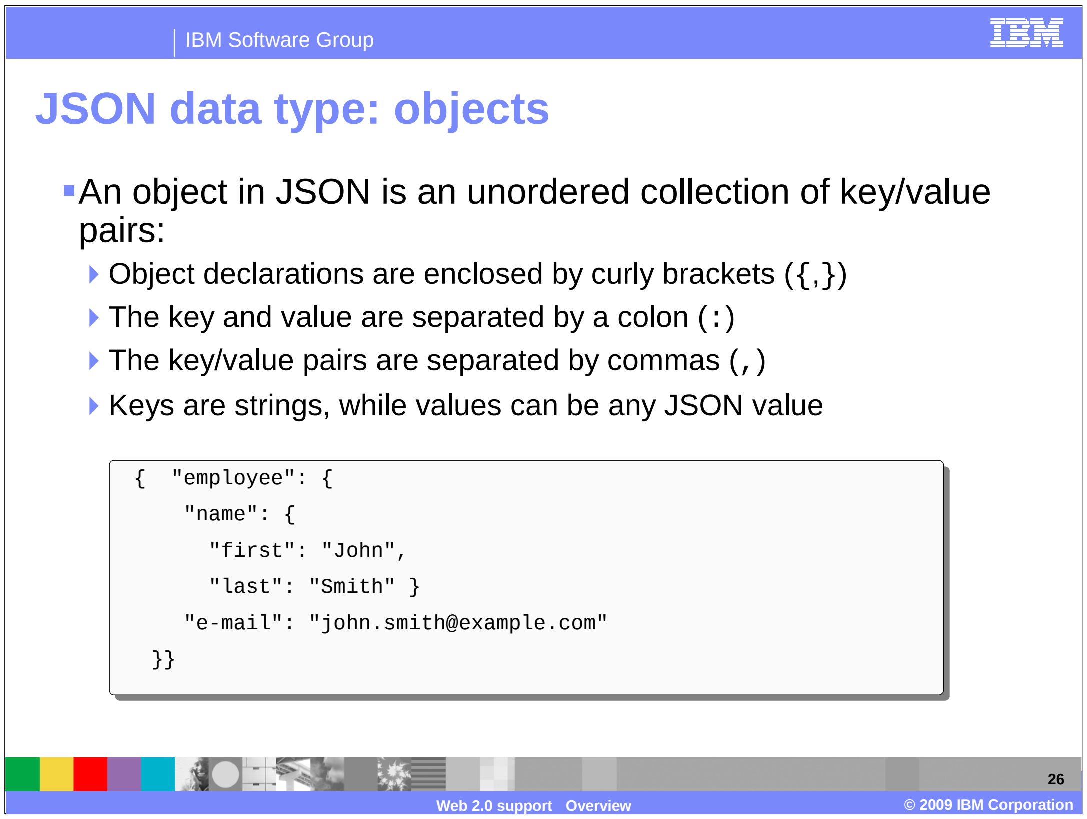
25

Web 2.0 support Overview © 2009 IBM Corporation

IBM has adopted the open-source Dojo toolkit as its internal standard. IBM is a key contributor to the Dojo project, and a committed member of the Dojo Foundation. IBM sees this as one of the most flexible of all the toolkits on the current market.

Dojo is an open source DHTML toolkit written in the JavaScript language. Dojo allows you to easily build dynamic capabilities into Web pages. You can use the components that Dojo provides to make your Web sites more usable, responsive, and functional. Dojo builds on several contributing code bases. “Unified” open source DHTML toolkit written in JavaScript, HTML and CSS. Dojo Toolkit works well across most modern browsers. Dojo provides a lot of power and attempts to make it digestible in three major layers: Dojo Core, Dijit, and DojoX. The Dojo DHTML toolkit provides an open-source framework that abstracts away the complexities of JavaScript development. It builds on several contributed code bases.

Dojo’s features include: A rich and extensible set of cross-browser, Ajax-based UI components or widgets, referred to as *dijits*. A powerful event handling system which extends the normal event handling model available in JavaScript. A simplified wrapper around Ajax functions, Drag-and-drop functionality, Internationalization, Data format conversion, HTML cookie handling, JSON support, and DOM manipulation to just name a few.

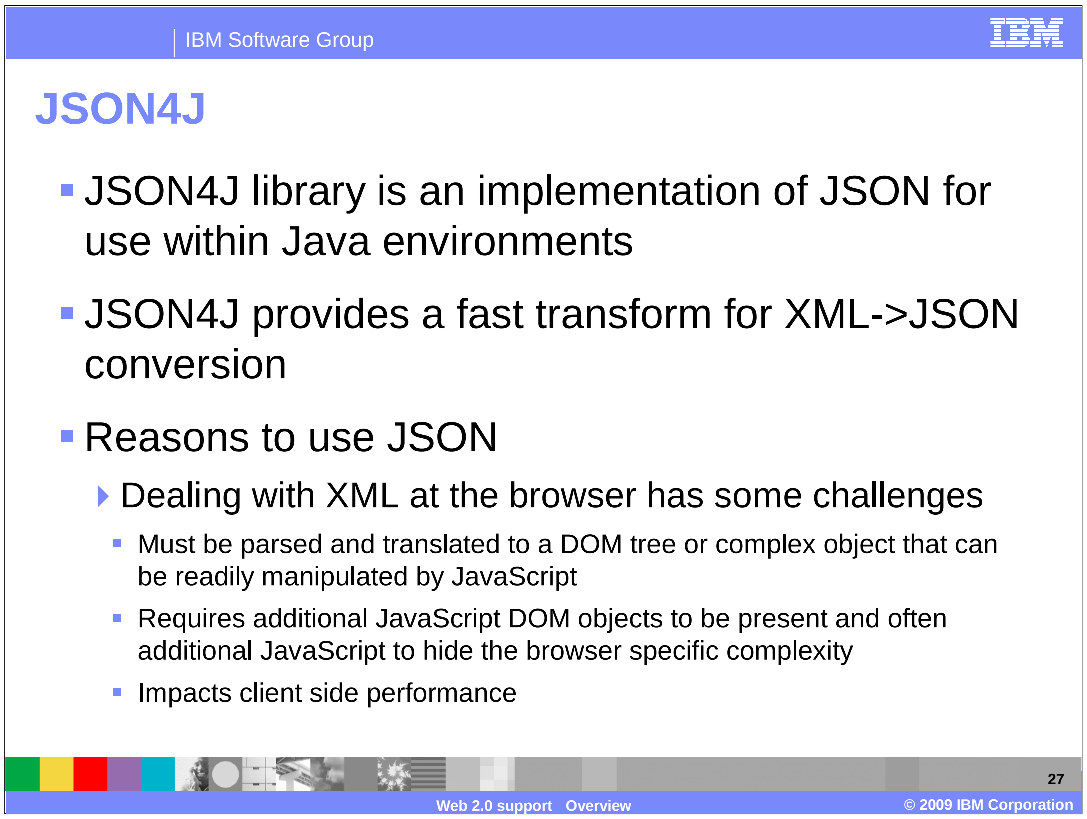


JSON data type: objects

- An object in JSON is an unordered collection of key/value pairs:
  - Object declarations are enclosed by curly brackets ({,})
  - The key and value are separated by a colon (:)
  - The key/value pairs are separated by commas (,)
  - Keys are strings, while values can be any JSON value

```
{  "employee": {
    "name": {
      "first": "John",
      "last": "Smith" }
    "e-mail": "john.smith@example.com"
  }}
```

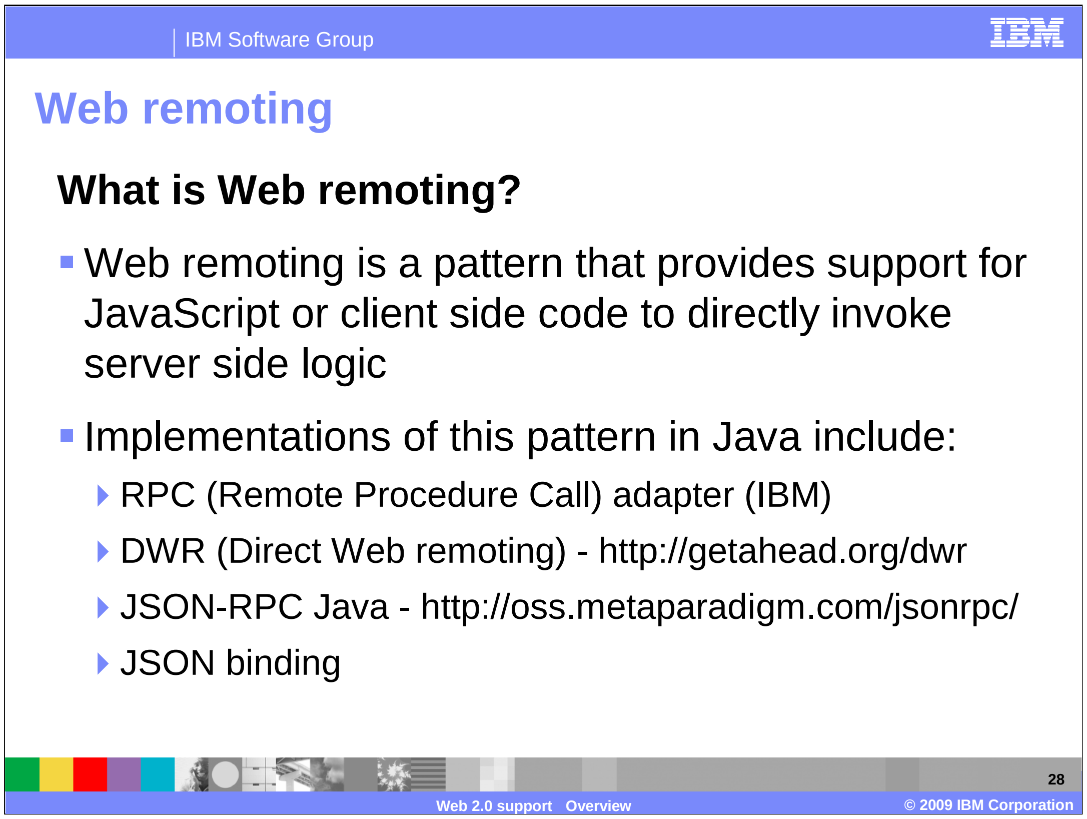
JSON Objects are created using the curly bracket, and employs name value pairs. Besides simple types, JSON objects can contain other JSON Object and JSON Arrays.



JSON4J

- JSON4J library is an implementation of JSON for use within Java environments
- JSON4J provides a fast transform for XML->JSON conversion
- Reasons to use JSON
  - Dealing with XML at the browser has some challenges
    - Must be parsed and translated to a DOM tree or complex object that can be readily manipulated by JavaScript
    - Requires additional JavaScript DOM objects to be present and often additional JavaScript to hide the browser specific complexity
    - Impacts client side performance

JSON4J library is an implementation of JSON for use within Java environments. JSON4J provides a fast transform for XML->JSON conversion. Dealing with XML at the browser has some challenges as it must be parsed and translated to a DOM tree or complex object that can be readily manipulated by JavaScript. XML requires additional JavaScript DOM objects (for example, XPathEvaluator or selectNodes) to be present and often additional JavaScript to hide the browser specific complexity which in turn Impacts client side performance.


IBM Software Group

Web remoting

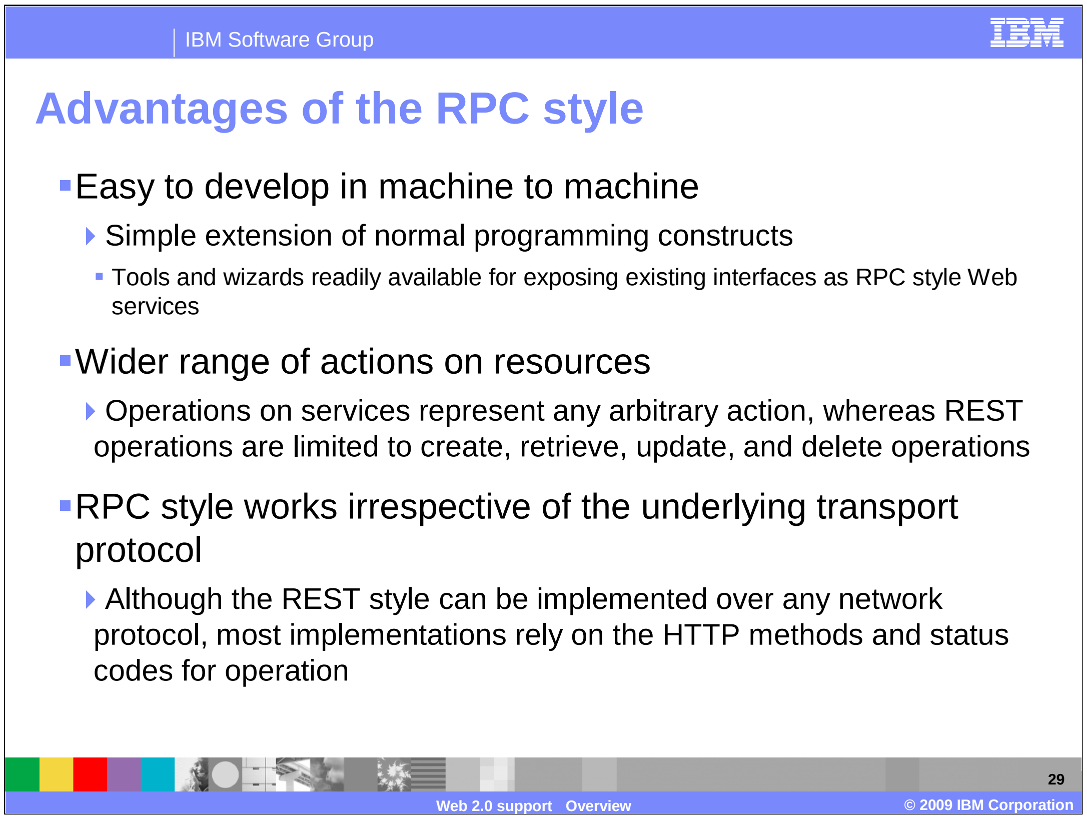
What is Web remoting?

- Web remoting is a pattern that provides support for JavaScript or client side code to directly invoke server side logic
- Implementations of this pattern in Java include:
  - RPC (Remote Procedure Call) adapter (IBM)
  - DWR (Direct Web remoting) - http://getahead.org/dwr
  - JSON-RPC Java - http://oss.metaparadigm.com/jsonrpc/
  - JSON binding

28

Web 2.0 support Overview © 2009 IBM Corporation

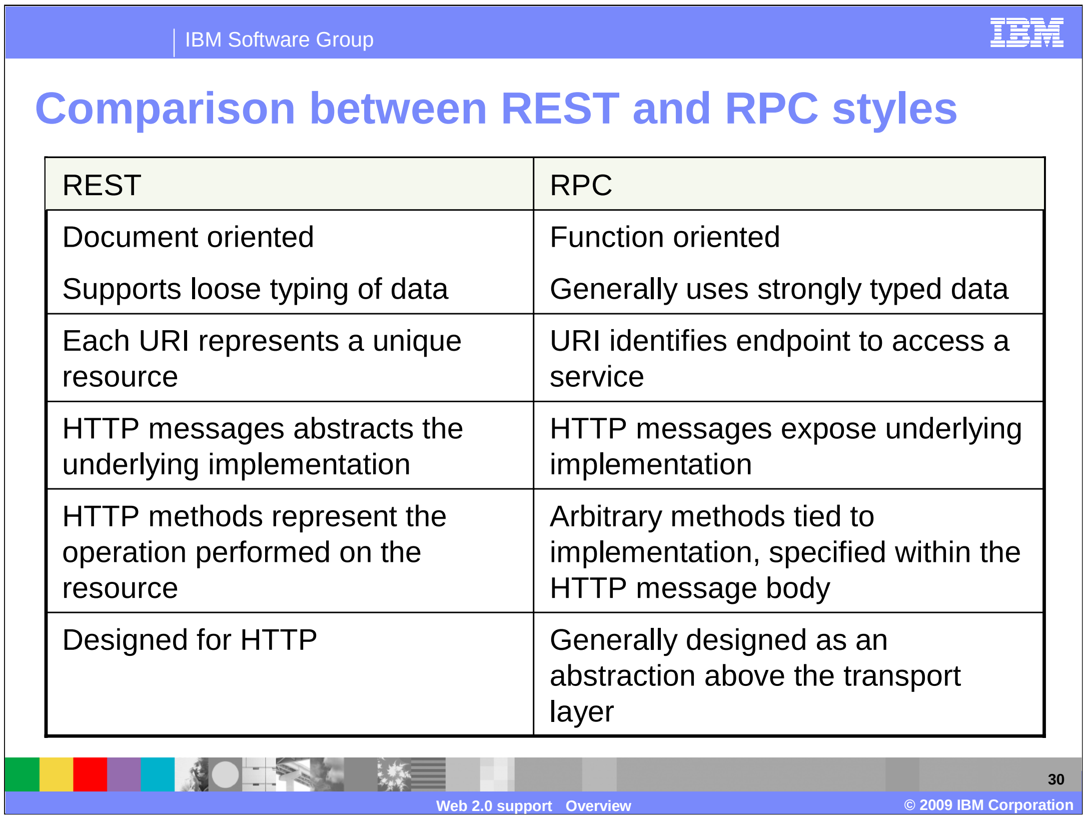
Web-remoting is a pattern that provides support for JavaScript or client side code to directly invoke server side logic. There are a few implementations of this pattern in Java. RPC (Remote Procedure Call) adapter is an IBM implementation for Web remoting to help developers create command-based services quickly and easily in a manner that complements programming styles for Ajax applications and other lightweight clients. DWR (Direct Web remoting) allows JavaScript in a browser to interact with Java on a server and helps you manipulate Web pages with the results. JSON-RPC-Java is a key piece of Java Web application middleware that allows JavaScript DHTML Web applications to call remote methods in a Java Application Server without the need for page reloading (Ajax). JSON data binding is used to transform JSON input arguments into Java types, and to transform Java results into JSON



Advantages of the RPC style

- Easy to develop in machine to machine
  - Simple extension of normal programming constructs
    - Tools and wizards readily available for exposing existing interfaces as RPC style Web services
- Wider range of actions on resources
  - Operations on services represent any arbitrary action, whereas REST operations are limited to create, retrieve, update, and delete operations
- RPC style works irrespective of the underlying transport protocol
  - Although the REST style can be implemented over any network protocol, most implementations rely on the HTTP methods and status codes for operation

RPC is an alternative to REST and has advantages as well. Tools and wizards readily available for exposing existing interfaces as RPC style Web services. This is largely due to its machine to machine interaction. Server programs using REST typically has to code to a lower level HTTP API and invoke URLs. In addition, not everything can naturally be represented RESTfully, using four verbs create, retrieve, update and delete. Some things are better represented as a set of operations, for example, transferFunds().



IBM Software Group

Comparison between REST and RPC styles

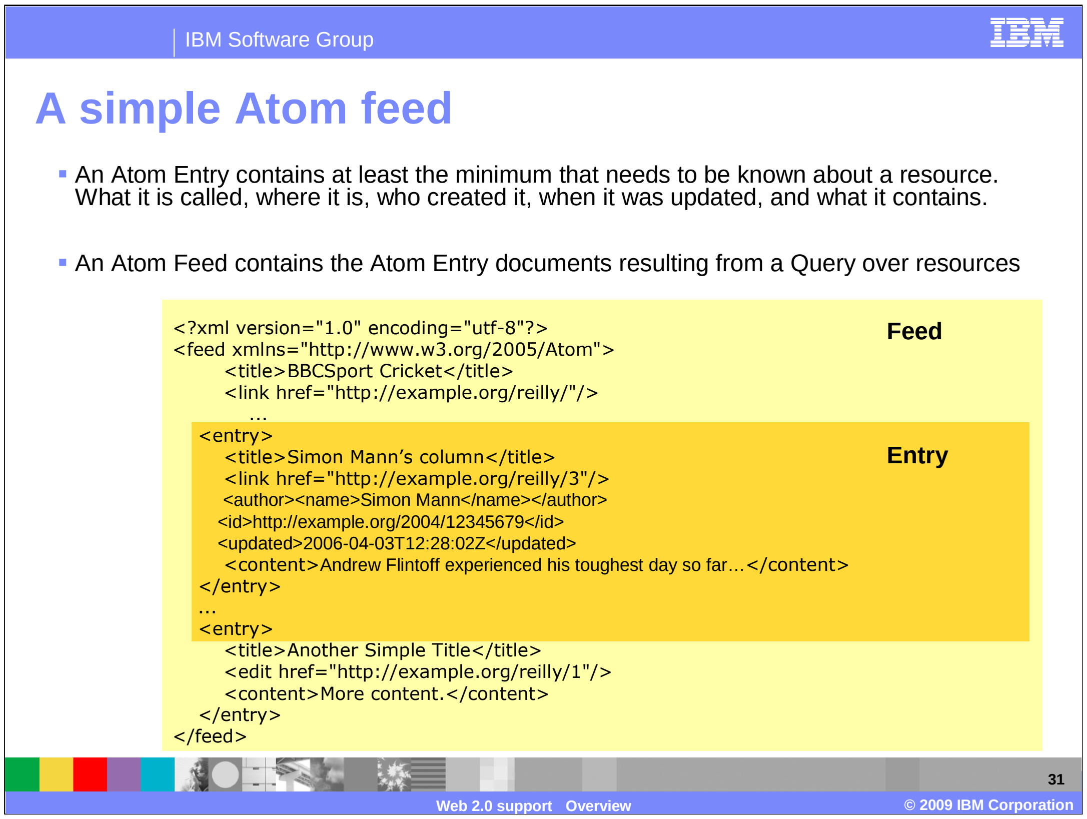
REST	RPC
Document oriented Supports loose typing of data	Function oriented Generally uses strongly typed data
Each URI represents a unique resource	URI identifies endpoint to access a service
HTTP messages abstracts the underlying implementation	HTTP messages expose underlying implementation
HTTP methods represent the operation performed on the resource	Arbitrary methods tied to implementation, specified within the HTTP message body
Designed for HTTP	Generally designed as an abstraction above the transport layer

30

Web 2.0 support   Overview

© 2009 IBM Corporation

The table above represents some of the key differences between REST Style interactions and RPC. RPC interactions are similar to SOAP and Web services in that there is a description document (such as WSDL) and the payload contains information about the operation. REST is about accessing a resources in a document oriented style and uses the HTTP Architecture.


IBM Software Group

A simple Atom feed

- An Atom Entry contains at least the minimum that needs to be known about a resource. What it is called, where it is, who created it, when it was updated, and what it contains.
- An Atom Feed contains the Atom Entry documents resulting from a Query over resources

```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">
       <title>BBCSport Cricket</title>
       <link href="http://example.org/reilly/"/>
          …
   <entry>
       <title>Simon Mann's column</title>
       <link href="http://example.org/reilly/3"/>
       <author><name>Simon Mann</name></author>
      <id>http://example.org/2004/12345679</id>
      <updated>2006-04-03T12:28:02Z</updated>
       <content>Andrew Flintoff experienced his toughest day so far…</content>
   </entry>
   …
   <entry>
       <title>Another Simple Title</title>
       <edit href="http://example.org/reilly/1"/>
       <content>More content.</content>
   </entry>
</feed>
```

Feed

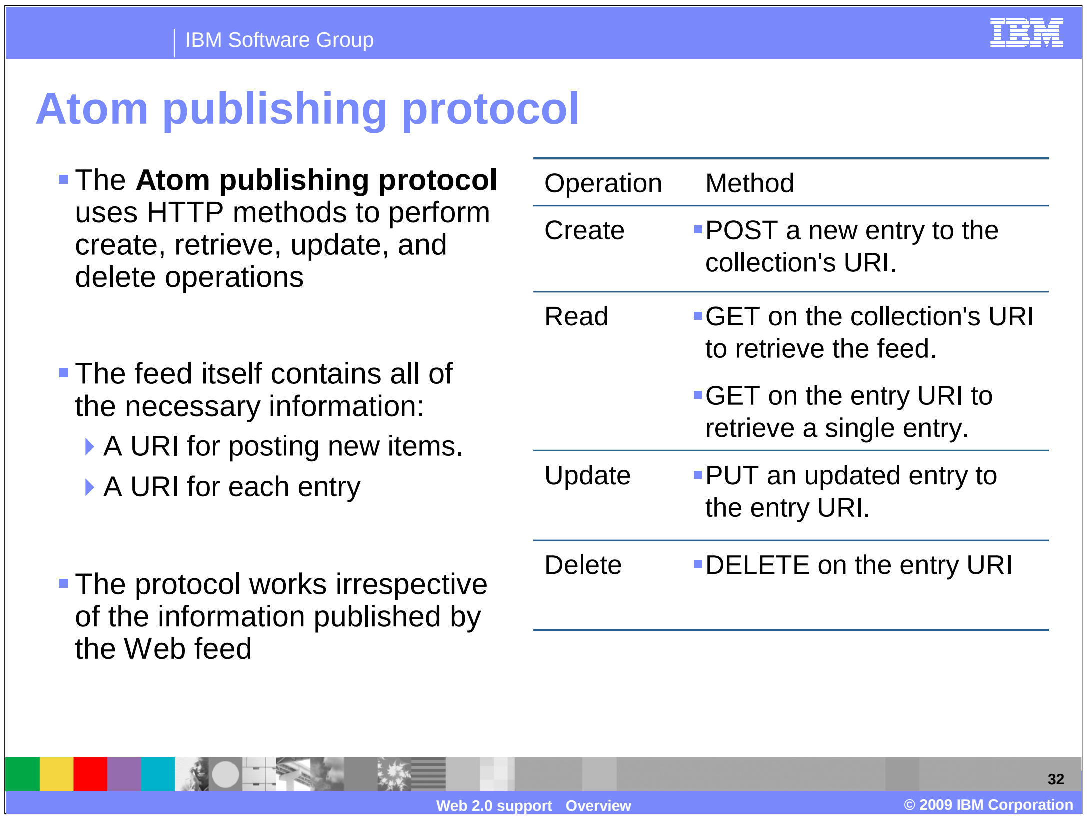
Entry

31

Web 2.0 support   Overview

© 2009 IBM Corporation

A simple Atom feed is shown above. Again, it is based off of XML. An Atom feed has two parts; the Atom Feed and the Atom Entry. An Atom Entry contains at least the minimum that needs to be known about a resource. The details of the resource include what it's called, where it is, who created it, when it was updated, and what it contains. An Atom Feed contains the Atom Entry documents resulting from a Query over resources.

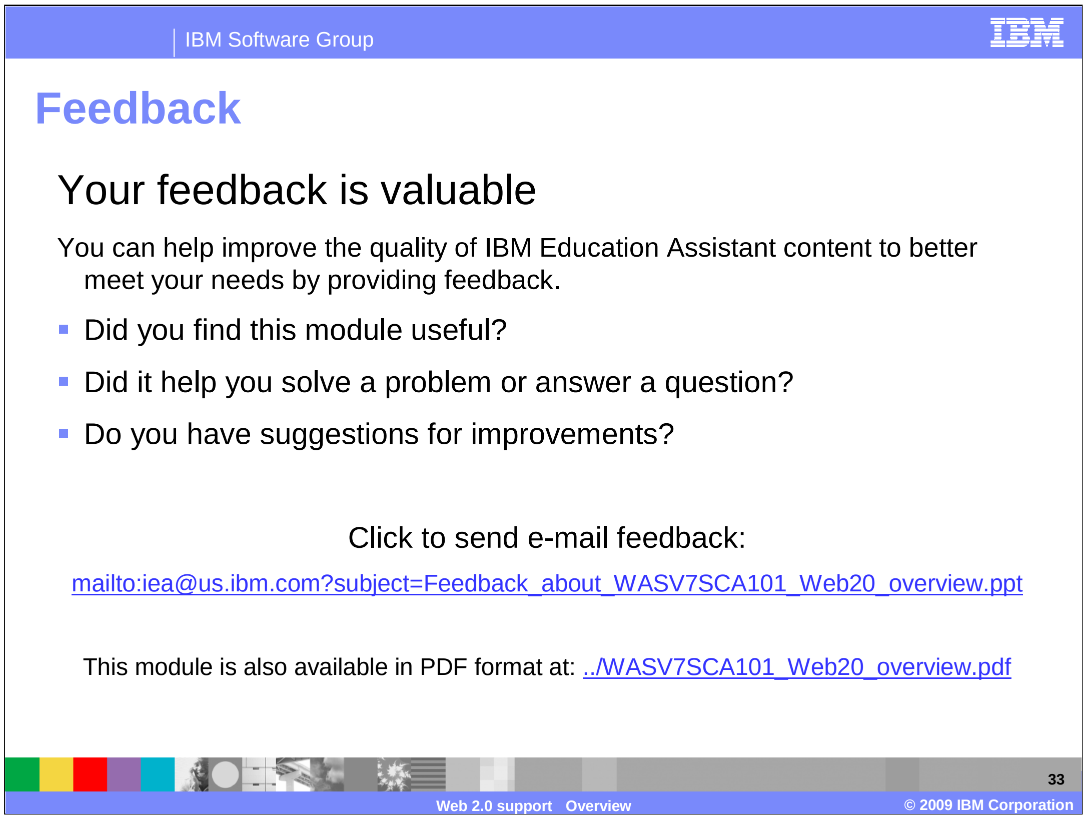


Atom publishing protocol

- The **Atom publishing protocol** uses HTTP methods to perform create, retrieve, update, and delete operations
- The feed itself contains all of the necessary information:
  - A URI for posting new items.
  - A URI for each entry
- The protocol works irrespective of the information published by the Web feed

Operation	Method
Create	POST a new entry to the collection's URI.
Read	GET on the collection's URI to retrieve the feed. GET on the entry URI to retrieve a single entry.
Update	PUT an updated entry to the entry URI.
Delete	DELETE on the entry URI

The Atom Publishing protocol extends Atom using the REST principal to allow for creation, retrieving, updating, and deleting of feeds. The feed itself contains all of the necessary information such as a URI for posting new items and a URI for each entry. The protocol works irrespective of the information published by the Web feed.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

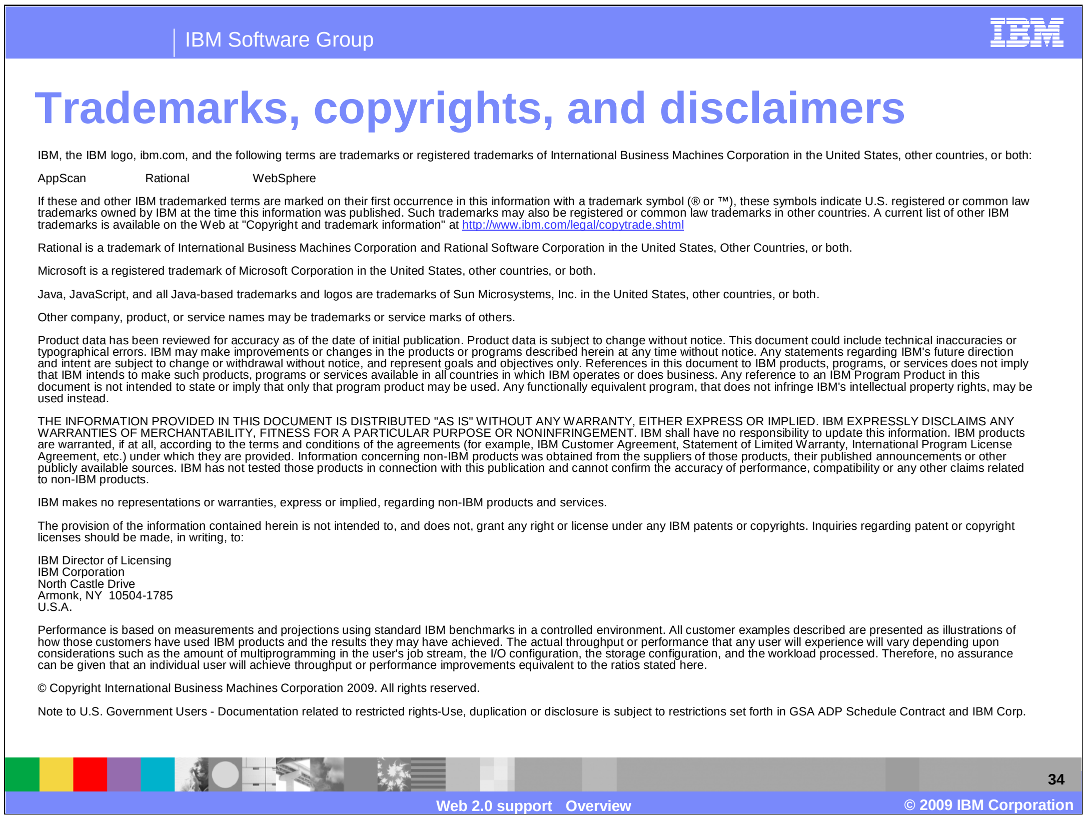
- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WASV7SCA101_Web20_overview.ppt

This module is also available in PDF format at: ../WASV7SCA101_Web20_overview.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers