Learn how to make your web pages talk and listen at the same time.

Get a handle on trees and the Document Object Model.

Transfer your data with plain text, XML, and JSON.

Learn how Sally did two things at the same time with asynchronous programming.

Make your clunky web apps feel like dynamic, responsive desktop applications.

Rebecca M. Riordan
Rob's Rock 'n' Roll Memorabilia

Meet Rob. He’s put all his savings into an online rock n’ roll memorabilia store. The site looks great, but he’s still been getting tons of complaints. Customers are clicking on the thumbnail images on the inventory page, but instead of showing information about the selected item, the customers’ browsers are taking forever to do anything. Some of Rob’s users are hanging around, but most have just stopped coming to Rob’s shop altogether.

This pane contains thumbnails of the item Rob has for sale.

When the user clicks an item, a bigger picture of the image is displayed here....

...and the details about the item are shown here.

I’m desperate... but I can’t afford a more powerful server, or a team of web experts.

Ajax pages only talk to the server when they have to... and only about what the server knows.

The problem with Rob’s site isn’t that his server is too slow... it’s that his pages are sending requests to the server *all the time*... even when they don’t need to.
Here’s what Rob’s online store does right now. What’s wrong with this picture?

The user clicks a thumbnail.

The browser sends the selected item’s ID to the server.

The server sends back a new page, with the selected item’s information.

The user clicks another thumbnail.

The browser sends the new item’s ID to the server.

The server sends back another whole new page.

The user gets tired of waiting and does something else...

How would Ajax change this diagram? Write down what you think should happen on Rob’s site.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________
Your job was to think about how Ajax could help save Rob's site... and his business. With Ajax, we can completely remove all page refreshes on his inventory page. Here's what that would look like:

**BULLET POINTS**

- Asynchronous requests allow **more than one thing** to happen at the same time.
- Only the part of a web page that needs to change gets updated.
- The page isn’t frozen while the user waits for the server to return.
Put a checkmark next to the benefits that you think Ajax can provide to your web applications.

- The browser can request multiple things from the server at the same time.
- Browser requests return a lot faster.
- Colors are rendered more faithfully.
- Only the parts of the page that actually change are updated.
- Server traffic is reduced.
- Pages are less vulnerable to compatibility issues.
- The user can keep working while the page updates.
- Some changes can be handled without a server round-trip.
- Your boss will love you.
- Only the parts of the page that actually change are updated.

Not all pages will reap every benefits of Ajax. In fact, some pages wouldn’t benefit from Ajax at all. Which of the benefits you checked off above do you think Rob’s page will see?
Browsers can operate asynchronously, but they usually make synchronous requests unless you indicate to them that they should do something differently.

- **The browser can request multiple things from the server at the same time.**
  
  This is only true sometimes. The speed of a request and response depends on what the server is returning. And it’s possible to build Ajax pages that are slower than traditional pages.

- **Browser requests return a lot faster.**
  
  Color rendering is dictated by the user’s monitor, not your app.

- **Colors are rendered more faithfully.**

- **Only the parts of the page that actually change are updated.**
  
  Reduced server traffic can save your users a lot of time, and your clients a lot of money on bandwidth.

- **Server traffic is reduced.**
  
  Because Ajax pages rely on technologies in addition to XHTML, compatibility issues can actually be a bigger problem with Ajax. Test, test, test your apps on the browsers your users have installed.

- **Pages are less vulnerable to compatibility issues.**
  
  Sometimes you want a user to wait on the server’s response, but that doesn’t mean you can’t still use Ajax. We’ll look at asynchrony more in Chapter 4.

- **The user can keep working while the page updates.**
  
  Handling things at the client can make your web application feel more like a desktop application.

- **Some changes can be handled without a server round-trip.**
  
  If you use Ajax in a way that helps your apps, the boss will love you. But you shouldn’t use Ajax everywhere… more on that later.

- **Your boss will love you.**

- **Only the parts of the page that actually change are updated.**

  Yes, this is the second time this shows up in the list. It’s that important!
Q: First you said Ajax was the web reinvented. Now it’s increasing server traffic. Which is it?

A: Sometimes it’s both! Ajax is one way to make requests, get responses, and build responsive web apps. But you’ve still got to be smart about when to make an asynchronous request, and when a regular synchronous request would be a better idea.

Q: How do I know when to use Ajax and asynchronous requests, and when not to?

A: Think about it like this: if you want something to go on while your user’s still working, you probably want an asynchronous request. But if your user needs information or a response from your app before they continue, then you want to make them wait. That usually means a synchronous request.

Q: So for Rob’s online store, since we want users to keep browsing while we’re loading product images and descriptions, we’d want an asynchronous request. Right?

A: Exactly. That particular part of Rob’s app—checking out different items—shouldn’t require the user to wait every time they select a new item. So that’s a great place to use Ajax and make an asynchronous request.

Q: And how do I do that?

A: Good question. Turn the page, and let’s get down to actually using Ajax to fix up Rob’s online store.