



Ajax

IN ACTION

Dave Crane
Eric Pascarello
with Darren James

contents

preface xix
acknowledgments xxi
about this book xxiv

PART 1 RETHINKING THE WEB APPLICATION 1

1 *A new design for the Web* 3

1.1 Why Ajax rich clients? 5

Comparing the user experiences 5 ▪ *Network latency* 9
Asynchronous interactions 12 ▪ *Sovereign and transient usage patterns* 15 ▪ *Unlearning the Web* 16

1.2 The four defining principles of Ajax 17

The browser hosts an application, not content 17 ▪ *The server delivers data, not content* 19 ▪ *User interaction with the application can be fluid and continuous* 21 ▪ *This is real coding and requires discipline* 23

1.3 Ajax rich clients in the real world 24

Surveying the field 24 ▪ *Google Maps* 25

1.4 Alternatives to Ajax 28

Macromedia Flash-based solutions 28 ▪ *Java Web Start and related technologies* 28

- 1.5 Summary 29
- 1.6 Resources 30

2

First steps with Ajax 31

- 2.1 The key elements of Ajax 32
- 2.2 Orchestrating the user experience with JavaScript 34
- 2.3 Defining look and feel using CSS 36
 - CSS selectors* 37
 - *CSS style properties* 39
 - A simple CSS example* 40
- 2.4 Organizing the view using the DOM 45
 - Working with the DOM using JavaScript* 47
 - *Finding a DOM node* 49
 - *Creating a DOM node* 50
 - *Adding styles to your document* 51
 - *A shortcut: Using the innerHTML property* 53
- 2.5 Loading data asynchronously using XML technologies 53
 - IFrames* 54
 - *XmlDocument and XMLHttpRequest objects* 56
 - Sending a request to the server* 58
 - *Using callback functions to monitor the request* 61
 - *The full lifecycle* 62
- 2.6 What sets Ajax apart 65
- 2.7 Summary 67
- 2.8 Resources 68

3

Introducing order to Ajax 69

- 3.1 Order out of chaos 71
 - Patterns: Creating a common vocabulary* 71
 - Refactoring and Ajax* 72
 - *Keeping a sense of proportion* 73
 - Refactoring in action* 73
- 3.2 Some small refactoring case studies 77
 - Cross-browser inconsistencies: Façade and Adapter patterns* 77
 - Managing event handlers: Observer pattern* 80
 - Reusing user action handlers: Command pattern* 83
 - Keeping only one reference to a resource: Singleton pattern* 87
- 3.3 Model-View-Controller 91
- 3.4 Web server MVC 93
 - The Ajax web server tier without patterns* 93
 - *Refactoring the domain model* 96
 - *Separating content from presentation* 100

- 3.5 Third-party libraries and frameworks 103
 - Cross-browser libraries* 104 ■ *Widgets and widget suites* 108
 - Application frameworks* 111
- 3.6 Summary 114
- 3.7 Resources 115

PART 2 CORE TECHNIQUES..... 117

- 4** ***The page as an application*** 119
 - 4.1 A different kind of MVC 120
 - Repeating the pattern at different scales* 120
 - Applying MVC in the browser* 122
 - 4.2 The View in an Ajax application 124
 - Keeping the logic out of the View* 124
 - Keeping the View out of the logic* 130
 - 4.3 The Controller in an Ajax application 134
 - Classic JavaScript event handlers* 134
 - The W3C event model* 137
 - Implementing a flexible event model in JavaScript* 138
 - 4.4 Models in an Ajax application 143
 - Using JavaScript to model the business domain* 144
 - Interacting with the server* 145
 - 4.5 Generating the View from the Model 147
 - Reflecting on a JavaScript object* 147 ■ *Dealing with arrays and objects* 151 ■ *Adding a Controller* 154
 - 4.6 Summary 157
 - 4.7 Resources 158
- 5** ***The role of the server*** 159
 - 5.1 Working with the server side 160
 - 5.2 Coding the server side 161
 - Popular implementation languages* 161
 - N-tier architectures* 162
 - Maintaining client-side and server-side domain models* 163

- 5.3 The big picture: common server-side designs 164
 - Naive web server coding without a framework* 164
 - Working with Model2 workflow frameworks* 166
 - Working with component-based frameworks* 167
 - Working with service-oriented architectures* 170
- 5.4 The details: exchanging data 174
 - Client-only interactions* 175
 - Introducing the planet browser example* 175
 - Thinking like a web page: content-centric interactions* 178
 - Thinking like a plug-in: script-centric interactions* 182
 - Thinking like an application: data-centric interactions* 188
- 5.5 Writing to the server 193
 - Using HTML forms* 193 ▪ *Using the XMLHttpRequest object* 195 ▪ *Managing user updates effectively* 197
- 5.6 Summary 206
- 5.7 Resources 207

PART 3 PROFESSIONAL AJAX..... 209

6 *The user experience* 211

- 6.1 Getting it right: building a quality application 212
 - Responsiveness* 213 ▪ *Robustness* 213 ▪ *Consistency* 214
 - Simplicity* 215 ▪ *Making it work* 215
- 6.2 Keeping the user informed 216
 - Handling responses to our own requests* 216
 - Handling updates from other users* 218
- 6.3 Designing a notification system for Ajax 222
 - Modeling notifications* 223
 - Defining user interface requirements* 225
- 6.4 Implementing a notification framework 226
 - Rendering status bar icons* 226 ▪ *Rendering detailed notifications* 229 ▪ *Putting the pieces together* 230
- 6.5 Using the framework with network requests 237
- 6.6 Indicating freshness of data 241
 - Defining a simple highlighting style* 241
 - Highlighting with the Scriptaculous Effects library* 243

6.7 Summary 244

6.8 Resources 245

7 **Security and Ajax 246**

7.1 JavaScript and browser security 247

Introducing the “server of origin” policy 248 ■ *Considerations for Ajax* 248 ■ *Problems with subdomains* 249
Cross-browser security 250

7.2 Communicating with remote services 251

Proxying remote services 252
Working with web services 253

7.3 Protecting confidential data 263

The man in the middle 263 ■ *Using secure HTTP* 264
Encrypting data over plain HTTP using JavaScript 266

7.4 Policing access to Ajax data streams 268

Designing a secure web tier 268
Restricting access to web data 272

7.5 Summary 277

7.6 Resources 278

8 **Performance 279**

8.1 What is performance? 280

8.2 JavaScript execution speed 281

Timing your application the hard way 282
Using the Venkman profiler 288
Optimizing execution speed for Ajax 289

8.3 JavaScript memory footprint 302

Avoiding memory leaks 302
Special considerations for Ajax 306

8.4 Designing for performance 311

Measuring memory footprint 312 ■ *A simple example* 316
Results: how to reduce memory footprint 150-fold 321

8.5 Summary 323

8.6 Resources 324

PART 4 AJAX BY EXAMPLE 325

9

Dynamic double combo 327

- 9.1 A double-combo script 328
 - Limitations of a client-side solution* 328
 - Limitations of a server-side solution* 329
 - Ajax-based solution* 330
- 9.2 The client-side architecture 331
 - Designing the form* 331
 - Designing the client/server interactions* 333
- 9.3 Implementing the server: VB .NET 334
 - Defining the XML response format* 335
 - Writing the server-side code* 336
- 9.4 Presenting the results 339
 - Navigating the XML document* 339
 - Applying Cascading Style Sheets* 342
- 9.5 Advanced issues 343
 - Allowing multiple-select queries* 343
 - Moving from a double combo to a triple combo* 345
- 9.6 Refactoring 345
 - New and improved net.ContentLoader* 346
 - Creating a double-combo component* 352
- 9.7 Summary 359

10

Type-ahead suggest 361

- 10.1 Examining type-ahead frameworks 362
 - Type-ahead suggest frameworks* 362 ■ *Google Suggest* 364
 - The Ajax in Action type-ahead* 365
- 10.2 The server-side framework: C# 366
 - The server and the database* 366
 - Testing the server-side code* 368
- 10.3 The client-side framework 369
 - The HTML* 369 ■ *The JavaScript* 370
 - Accessing the server* 380

- 10.4 Adding functionality: multiple elements with different queries 392
- 10.5 Refactoring 392
 - Day 1: developing the TextSuggest component game plan* 394
 - Day 2: TextSuggest creation—clean and configurable* 397
 - Day 3: Ajax enabled* 401 ▪ *Day 4: handling events* 406
 - Day 5: the suggestions pop-up UI* 413
 - Refactor debriefing* 421
- 10.6 Summary 422

11 **The enhanced Ajax web portal 423**

- 11.1 The evolving portal 424
 - The classic portal* 424 ▪ *The rich user interface portal* 426
- 11.2 The Ajax portal architecture using Java 427
- 11.3 The Ajax login 429
 - The user table* 429 ▪ *The server-side login code: Java* 430
 - The client-side login framework* 433
- 11.4 Implementing DHTML windows 439
 - The portal windows database* 439
 - The portal window's server-side code* 441
 - Adding the JS external library* 445
- 11.5 Adding Ajax autosave functionality 448
 - Adapting the library* 448
 - Autosaving the information to the database* 450
- 11.6 Refactoring 453
 - Defining the constructor* 455 ▪ *Adapting the AjaxWindows.js library* 456 ▪ *Specifying the portal commands* 458
 - Performing the Ajax processing* 462
 - Refactoring debrief* 464
- 11.7 Summary 464

12 **Live search using XSLT 466**

- 12.1 Understanding the search techniques 467
 - Looking at the classic search* 467 ▪ *The flaws of the frame and pop-up methods* 469 ▪ *Examining a live search with Ajax and XSLT* 470 ▪ *Sending the results back to the client* 472

- 12.2 The client-side code 473
 - Setting up the client* 473
 - Initiating the process* 474
- 12.3 The server-side code: PHP 476
 - Building the XML document* 476
 - Building the XSLT document* 479
- 12.4 Combining the XSLT and XML documents 481
 - Working with Microsoft Internet Explorer* 483
 - Working with Mozilla* 484
- 12.5 Completing the search 485
 - Applying a Cascading Style Sheet* 485
 - *Improving the search* 487
 - *Deciding to use XSLT* 489
 - Overcoming the Ajax bookmark pitfall* 490
- 12.6 Refactoring 491
 - An XSLTHelper* 492
 - *A live search component* 496
 - Refactoring debriefing* 501
- 12.7 Summary 501

13

Building stand-alone applications with Ajax 503

- 13.1 Reading information from the outside world 504
 - Discovering XML feeds* 505
 - Examining the RSS structure* 506
- 13.2 Creating the rich user interface 509
 - The process* 510
 - *The table-less HTML framework* 511
 - Compliant CSS formatting* 513
- 13.3 Loading the RSS feeds 518
 - Global scope* 518
 - *Ajax preloading functionality* 520
- 13.4 Adding a rich transition effect 524
 - Cross-browser opacity rules* 524
 - *Implementing the fading transition* 525
 - *Integrating JavaScript timers* 527
- 13.5 Additional functionality 528
 - Inserting additional feeds* 529
 - Integrating the skipping and pausing functionality* 531

13.6	Avoiding the project's restrictions	534
	<i>Overcoming Mozilla's security restriction</i>	534
	<i>Changing the application scope</i>	537
13.7	Refactoring	537
	<i>RSS reader Model</i>	537
	▪ <i>RSS reader view</i>	541
	<i>RSS reader Controller</i>	545
	▪ <i>Refactoring debrief</i>	558
13.8	Summary	559
appendix A	<i>The Ajax craftsperson's toolkit</i>	561
A.1	Working smarter with the right toolset	562
A.2	Editors and IDEs	565
A.3	Debuggers	571
A.4	DOM inspectors	582
A.5	Installing Firefox extensions	585
A.6	Resources	588
appendix B	<i>JavaScript for object-oriented programmers</i>	589
B.1	JavaScript is not Java	590
B.2	Objects in JavaScript	592
B.3	Methods and functions	606
B.4	Conclusions	617
B.5	Resources	617
appendix C	<i>Ajax frameworks and libraries</i>	619
	<i>index</i>	635