ADOBE®

FLASH® PLATFORM FROM START TO FINISH

WORKING COLLABORATIVELY USING **ADOBE® CREATIVE SUITE® 5**

AARON PEDERSEN

JAMES POLANCO

DOUG WINNIE

Adobe

Adobe Flash Platform from Start to Finish:
Working Collaboratively Using Adobe Creative Suite 5
Aaron Pedersen, James Polanco, and Doug Winnie

This Adobe Press book is published by Peachpit.

Peachpit
1249 Eighth Street
Berkeley, CA 94710
510/524-2178
510/524-2221 (fax)

Peachpit is a division of Pearson Education.
For the latest on Adobe Press books, go to www.adobepress.com
To report errors, please send a note to errata@peachpit.com

Copyright © 2010 by Aaron Pedersen, James Polanco, and Doug Winnie

Acquisitions Editor: Victor Gavenda
Project Editor: Susan Rimerman
Production Editor: Cory Borman
Developmental Editor: Kim Wimpsett
Copy Editor: Cathy Caputo
Cover Design: Mimi Heft
Interior Design and Composition: Kim Scott, Bumpy Design
Indexer: James Minkin

Notice of Rights
All rights reserved. No part of this book may be reproduced or transmitted in any form by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. For information on getting permission for reprints and excerpts, contact: permissions@peachpit.com.

Notice of Liability
The information in this book is distributed on an "As Is" basis, without warranty. While every precaution has been taken in the preparation of the book, neither the authors nor Peachpit shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the instructions contained in this book or by the computer software and hardware products described in it.

Trademarks
Adobe, Flash, Flash Catalyst, Flex Builder, and Creative Suite are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. All other trademarks are the property of their respective owners.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Peachpit was aware of a trademark claim, the designations appear as requested by the owner of the trademark. All other product names and services identified throughout this book are used in editorial fashion only and for the benefit of such companies with no intention of infringement of the trademark. No such use, or the use of any trade name, is intended to convey endorsement or other affiliation with this book.

ISBN-13: 978-0-321-68071-6
ISBN-10: 0-321-68071-5

9 8 7 6 5 4 3 2 1

Printed and bound in the United States of America

Contents

PART I ▪ OVERVIEW

1 Why Projects Fail 3

What Failure Looks Like 4

State of the Three Ts: Technology, Tools, and Teams. 4

How to Deliver Successful Projects. 5

2 The Project Spectrum. 7

Small Projects. 8

Medium Projects 9

Large Projects. 10

Not All Projects Are the Same 11

3 Understanding Roles in a Project. 13

External Roles 14

Client. 14

Management. 14

Design and Development 14

Design 14

Development 17

Putting the disciplines together. 18

Support 18

Quality assurance. 18

Build and release 19

Getting Roles to Work Together 19

Defining a method for collaboration 19

Using the DACI 20

4 Defining Project Phases 21

Planning. 22

Design 23

Development 24

Build and Release. 24

Maintenance 25

PART II ▪ PLANNING

5 High-Level Vision and Team Organization 29

High-Level Vision 30

Who is the project targeting? 30

What is the business problem? 30

How should the problem be solved? 30

Why should it be solved this way? 31

When should it be solved? 31

Team Definition 32

Defining stakeholders 32

Determining team roles in the kick-off meeting 32

Scope Review and Definition 33

Understanding constraints 33

Conducting user research 34

Creating use cases 35

Feature Set Definition 35

Feature breakdown 36

Prioritization and dependencies 36

Initial estimates and next steps 37

Summary Checklist 37

6 Setting Expectations for Your Project 39

Building a Project's Specification. 40

Getting your team involved 40

Exploring the types of specifications 40

Creating a specification using Adobe Buzzword 41

Discovering Team Roles 43

Creating a contract between design and development . 43

Developing a Project Plan 45

Overview of project plans 45

Project plan formats and tools 45

Summary Checklist 48

7 Tuning and Adjusting for Success 49
Choosing the Right Technologies and Tools 50
Is your publishing size important? 50
How video-rich is your product? 51
How data-centric is your application? 51
What deployment environment is suitable for your application? 52
Adjusting Your Project 54
Refining the feature set. 54
Modifying project scope 54
Adapting team roles 54
Assigning the Order of Tasks: Waterfall vs. Iterative . . . 55
Waterfall 55
Iterative 56
Summary Checklist 57

PART III ▪ DESIGN

8 Planning Design 61
Defining the Intent of Design. 62
Identifying the emotion 62
Picturing the emotion 63
Documenting the emotion and action 63
Defining rules to design 65
Giving Your Creativity Structure. 67
Architecting the interface 68
Don't just take our word for it. 71
Syncing with your team 71
Summary Checklist 72

9 Iterative Design. 73
Adding Details to Your Design 74
Mapping the user flow 74
Creating your linked wireframes 76
Designing the Visuals 78
Designing through wireframes and comps 78
Working efficiently with Creative Suite design tools . . 88
Preparing for the Handoff. 94
Keeping a tidy project 94

Handing off to developers 97
Syncing with your team 99
Summary Checklist 100

PART IV ▪ DEVELOPMENT

10 Planning Development 103
Architecting Your Application 104
Use design patterns to solve common business problems 104
Consider adopting a framework 104
Architecting Your Components 106
Views 106
Components 106
Flex 4 skins 107
Architecting Application Content 108
Architecting the Server Communication Layer 109
What communication formats are available within the Flash Platform?. 110
Developing a data contract 114
Establishing coding standards 116
Syncing with your Team. 118
Summary Checklist 118

11 Iterative Development 119
Developing Your Initial Project. 120
Defining project structure based on your Flash Platform tool 120
Expanding your initial project structure. 126
Establishing an application testing harness 128
Distributing the project to team members. 129
Developing Components and Integrating Designs. . . . 129
Executing the designer and developer contract 130
Developing supporting tiers 134
Integrating design into components 134
Integrating components into the application 135
Iterating on components 135
Syncing with Your Team 136
Summary Checklist 137

PART V ▪ BUILD AND RELEASE

12 Planning Build and Release 141

Defining an Environment 142
 Determining your deployment 142
 Structuring your environments 143
Defining the Build Process 146
 Who is building and deploying? 146
 Do you need a dedicated process or team?. 147
 Do you need a build tool? 147
 Quality assurance standards 147
 Automated testing 149
Defining Application Performance Benchmarks 150
 Frames per second 150
 File size. 150
 Load time 150
 Perceived time. 151
 Memory usage. 151
 Minimum system requirements. 151
Summary Checklist 152

13 Iterative Build and Release 153

Creating Builds, Quality Assurance, and Deployment . 154
 Understanding the different types of testing 154
 Using a bug scrub process 157
 Defining your testing process 158
Using Flash Platform Features for Testing 159
 Debug vs. release builds 159
 Browser and server caching 159
 Runtime shared libraries. 160
 Local shared objects 161
 AIR application data storage 161
 AIR versioning 163
 Signing builds 163
Summary Checklist 164

14 Deploying Your Finished Project 165

Creating and Securing Your Final Build. 166
 Understanding AIR certificates 166
 Installing an AIR badge 167

Verifying Your Finished Project 168
Verification testing 168
Stress and performance testing 169
Looking Back on Your Project 169
Gathering your team 169
What worked? 170
What didn't work? 170
What can be fixed? 171
Summary Checklist 171

PART VI ▪ MAINTENANCE

15 Maintaining Projects After Deployment 175
Understanding Technology Impacts 176
Flash Player versions 176
AIR certificate updates 177
Flex SDK and RSL versions 178
Third-party library updates 178
Fixing Issues After Deployment 179
Critical fixes 179
Grouping issues 180
Making Changes and Adding Features 180
Making design changes 180
Making content updates 181
Adding new functionality 181
Leveraging source control 181
Recognizing End of Life 182
Summary Checklist 183

Glossary. 185

Index 189

Introduction

Adobe provides a deep ecosystem based around Flash Player. This ecosystem has been coined the *Flash Platform*, and it consists of a series of Adobe tools, technologies, and services.

Creating projects for the Flash Platform is an exciting process, yet it is often a daunting experience for everyone involved. Adobe is constantly evolving the platform to provide new and powerful features that not only give creators more control but also provide the ability for team members to work together more efficiently. Adobe Creative Suite 5 has introduced many new features and improvements to help teams work together so that the final product, be it a Web site, a banner ad, a desktop application, or a mobile game, is the best that it can be. By taking advantage of both Creative Suite 5 and the Flash Platform, project teams have the potential to build amazing applications, yet tools alone cannot guarantee that a project will be successful.

In this book, we examine why projects are not always successful and how to avoid many of these issues. We will look at key areas that can breed both success and the potential for failure. You will learn how to plan and execute all aspects of your project, from initial concept to the time your users no longer need your application. We will also spend a lot of time showing you cool new (and existing) features in Creative Suite 5 and the Flash Platform that can help improve your creation experience to give your project the power to succeed.

WHO THIS BOOK IS FOR

This book is for the entire project team that is looking to adopt and build projects using Adobe's CS5 and Flash Platform technologies. This includes teams of one to teams that span departments and companies. Our goal is to provide information about the Flash Platform project creation process for everyone on your team. This includes executive management, project management, design, development, testing, and deployment.

One of the keys to success is making sure that everyone involved with a project, including your customers, are "on the same page" throughout the entire process. If the entire process is not understood, then these unknown areas pose the most risk to hindering your project's success. We hope to shed light on all of these areas so that you and your team are prepared as much as possible.

WHO THIS BOOK ISN'T FOR

In this book we cover a lot of different and important subjects, yet we could not write a book that covers every topic needed for every person who wants to use the Flash Platform. If you are looking learn how to design and develop Flex and Flash for the first time, this is not the book for you. We make the assumption that you and your team are familiar with creating Flash-based content.

If you are looking for a book that examines a specific Creative Suite 5 tool or Flash Platform technology in-depth, then this book is not for you. We cover a lot of the tools and technologies as deeply as possible, but our goal is to focus more on highlighting important features and processes to help your team succeed, not master one specific tool.

If you are looking to learn the basics of Creative Suite 5, then this book is not for you. Similar to Flash and Flex, we cover a lot of ground, but we are assuming that you are familiar with using many of the tools included with Creative Suite.

HOW THIS BOOK IS ORGANIZED

Because we are writing a book that covers the Flash Platform project process from beginning to end, we have broken the book down into six phases: Overview, Planning, Design, Development, Build and Release, and Maintenance.

- **Overview.** We examine why projects fail, what makes up a project, who is involved in your project, and how the project proceeds over time.
- **Planning.** We cover the overall process for how to plan and execute your project, organize your requirements, set expectations for everyone involved, and make sure everyone is in agreement throughout the entire life of your project.
- **Design.** We discuss how to plan for design, who is involved in the design process, what the design team requires, and how to collaborate seamlessly with everyone involved in the project.
- **Development.** We talk about the planning process for development, who is involved in the development process, what the development team requires, and how to collaborate seamlessly with everyone involved in the project.
- **Build and Release.** We examine what the build and release process is, who is involved, how to plan for build and release, and how this process is important for guaranteeing the quality of your project.
- **Maintenance.** Just because your project has launched doesn't mean that you and your team's involvement is done. We look at common issues that arise post-release and how to support your project's growth and improvement over time.

It is important to note that we have written the chapters in a linear order, but we realize that many of these phases will overlap during the actual project. This is especially true for those teams that employ iterative processes such as Agile or Extreme Programming. Throughout the book we will examine how each phase interacts and often overlaps with other phases.

In each chapter, you will be presented with terms that are relative to the topic at hand. We have compiled a list of definitions of these terms in the glossary at the end of the book.

HOW TO CONTACT US

If you have questions about the processes outlined in the book or are looking for more details, we recommend checking out our Web sites.

You can find James and Aaron's blog at *http://www.developmentarc.com*. You can find Doug's blog at *http://www.adobe.dougwinnie.com*. You can also follow all three of the authors on Twitter:

Aaron: @aaronpedersen
James: @jamespolanco
Doug: @sfdesigner

This page intentionally left blank

CHAPTER 4

Defining Project Phases

Not every project is the same, but from a bird's-eye view, every project will consist of the same five phases: planning, design, development, build and release, and maintenance. This chapter will review the overall process at a high level. In subsequent parts of the book, we'll discuss each phase in more depth, including best practices for using Adobe software and technologies to facilitate the process.

Figure 4-1 shows the general workflow that we recommend for all your Flash Platform projects. Of course, the details and tasks that take place in each of these phases will vary based on the complexity and scope of your project, and we will cover those details in future chapters.

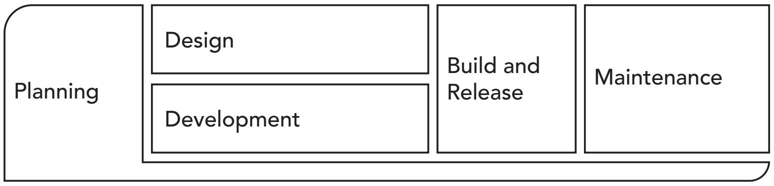

Figure 4-1
The overall workflow

As you can see, this process is fairly linear. The workflow covered in this book isn't focused on the methodology of the development process; instead, it's important to understand that each phase has a start and end, but certain ones take place concurrently with others. You may have heard people in the industry talk about iterative development models that aren't linear, such as Extreme Programming (XP), Agile, and others. The approach in this book consists of aspects of each of these methodologies and tells you how to facilitate them using Adobe tools, services, and technologies.

Now we'll discuss each section of the workflow in more detail and explore the goals that are associated with each of them.

PLANNING

During the planning phase of your project (**Figure 4-2**), you are defining what your project will require in order to be successful. This phase consists of several tasks, including constructing your project's high-level vision, determining the overall scope of your project, and, after analyzing the details of this scope, figuring out what type of resources you'll need in order to successfully complete the project based on a defined schedule.

Based on the scope of your project, the opinion of external stakeholders, or the requests of the client or customer, you may need to define specifically what the deliverables are going to be for the project. Good planning goes beyond the initial scope of the project to help facilitate long-term vision. However, it's important to keep deadlines and budgets in mind when drafting the project's scope. During the planning process, it's a good idea to keep your client involved and allow them to become as much of an expert on the feature details as you and your team.

Ultimately, you need to ensure that your client, customer, team, and stakeholders know exactly what will define success for the project. There should be no doubt at the end of the planning process on what success means or the steps that will be taken to ensure that the project is successful.

As you may recall from earlier chapters, the workflow covered in this book involves planning throughout the entire workflow, as indicated in Figure 4-2. Consistently reevaluating your execution on the project and making planning adjustments are a core part of the methodology.

WHO'S WHO IN THE PLANNING PHASE

Generally, this phase involves the business development team, project managers, and design and development leads, who help provide implementation and resource details.

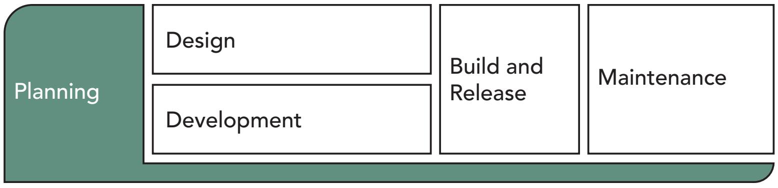

Figure 4-2
The planning phase

The chapters in Part II, "Planning," discuss this phase in more detail. We will discuss how a project can be taken from a simple idea and turned into a well-organized plan that can start your project on the road to success.

DESIGN

Most of the work that designers do involves putting their creative vision into words and attempting to communicate that vision as effectively as possible. Communicating something in words that is meant to stimulate all of the senses can be extremely difficult and time-consuming, but it's is also critical to the overall success of your project, especially when working with external development teams or vendors.

Each step of the design phase (see **Figure 4-3**) involves a creative process that evolves into a more detailed and granular expression of the overall project. This first starts with the creative brief that details the overall creative vision, theme, and objective of the visual design. Many times, the creative brief includes some user research to help back up the decision made on the vision, and it references other projects, clients, or competitors to help build a better understanding of what the project is going to do and, sometimes, what the project is meant to avoid.

Subsequent milestones for the design process can vary based on the complexity of your project and what your overall role is. If you are doing a lot of the creative work yourself, you want to carefully guide your creative process so you can get valuable feedback from your client, without a significant amount of throwaway work. Developing wireframes and prototypes can improve the collaborative design process with your client and designer. These valuable steps give you the ability to experiment and interpret your project design requirements while in an isolated environment.

When in these collaborative and experimental phases, it is important to note that two types of design can be in play at the same time. The first is the overall visual design of the project. This is the overall creative expression using a surface as the stage. The other complementary design discipline is user experience (UX) design, which defines how the user can interact with that surface to get a deeper level of immersion with the creative experience. Many people see these two disciplines at odds, but they must be seen as complementary design requirements for your project—one to design the surface and the other to make that surface personal.

After you define the design requirements of the project, you then need to communicate this as thoroughly as possible to ensure that the production and development teams can implement and honor the approved design as seamlessly as possible. Creating style guides, design comps, and assets can take the question out of how to apply design to the project. It's almost impossible to accommodate all situations or use cases, but with experience, this process becomes more efficient and streamlined.

The chapters in Part III, "Design," cover this phase in more detail. We will discuss the Flash Platform tools that can make this design process easier and how to create detailed wireframes and design comps.

WHO'S WHO IN THE DESIGN PHASE

Generally, this phase involves the creative side of the team with oversight from design and project management, who also keep the development team informed.

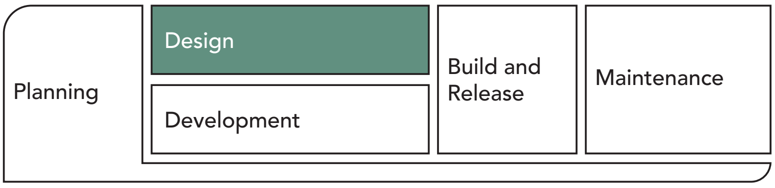

Figure 4-3
The design phase

DEVELOPMENT

Development starts with planning activities that give architects or lead developers the opportunity to examine and design a solid architecture that takes into account the feature set that was defined during the initial planning phase. This includes researching frameworks the project can leverage, syncing up with the server-side developers, and constructing the data schemas needed to communicate between the front-end and back-end applications. The outcome of this planning is to help provide feedback and detailed development estimates to the project management team so that they can construct a more precise project plan. Once you've planned the development, you can start the actual development of the product.

However, as you can see in **Figure 4-4**, development and design are parallel activities. This is because they continue to feed and consume each other in a symbiotic relationship. It is important as you continue through the development process that you are continually checking in with your design counterparts. This book refers to this as *syncing with your team* and the *designer and developer contract*.

Team sync includes regular meetings during the design and development phases to coordinate the parallel tasks at hand and to discuss the developer and designer contract. This contract is an informal guide on how to break up elements of an application or feature into small parts that can be defined by both design and code. This agreed-upon definition allows for a more seamless handoff when designs need to be integrated into the application code base.

The chapters in Part IV, "Development," cover this phase in more detail. We will discuss now to properly plan your development effort and how to successfully execute the construction of your application.

BUILD AND RELEASE

The build and release process (**Figure 4-5**) is broken down into four core aspects: development, testing, integrating, and distribution. The *development* aspect of the process is making sure that your development teams have a unified process for creating, managing, and verifying their content.

Testing is the process of making sure that the application your team is making behaves as intended and that no errors occur during normal (and ideally abnormal) use of the application.

WHO'S WHO IN THE DEVELOPMENT PHASE

Generally, this phase involves the programmers, database administrators, and software engineers of the team, with oversight from development leads and project management, who also keep the design team informed.

WHO'S WHO IN THE BUILD AND RELEASE PHASE

Generally, the build and release phase consists of team members responsible for testing and publishing versions of the project application for review and final release to the users.

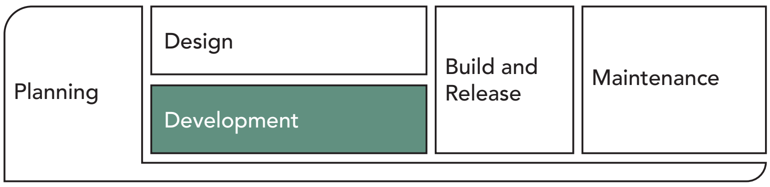

Figure 4-4
The development phase

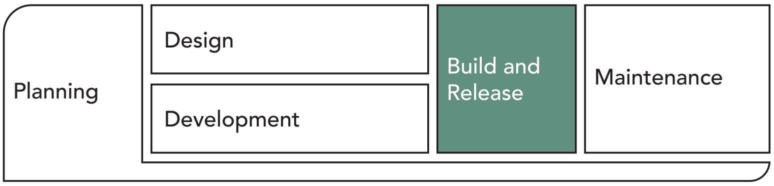

Figure 4-5
The build and release phase

Integration is the process of bringing together different team members' (or roles') work into a unified application. This can include bringing the final design into the application, merging different developers' code, or integrating live data feeds from the back-end server team into the final application.

The final aspect of build and release is *distribution*, which is the process of making the final application available to the end user. This could be posting the application on your Web site or having the IT team push the desktop application across the network.

Trying to determine who exactly should be involved in build and release and what roles they take on depends on the size of the project, the structure of your team, the organization of your company, what other companies are involved (if there are any), and of course your client.

For larger projects, a dedicated staff, either internal or a third party, may handle all of the build and release roles. For smaller teams and projects, build and release roles may be assigned to your developers, designers, management, and even the client depending on your needs and requirements.

The chapters in Part V, "Build and Release," cover this phase in more detail. We will discuss what the build and release process entails and how your team can manage this process.

MAINTENANCE

Many projects, especially the medium-size to large ones, have long life spans that extend well beyond the first release of the application. Other projects, usually smaller ones, may not require any maintenance post-release. The maintenance phase (**Figure 4-6**) encompasses this post-release process and can include adding functionality, fixing newly discovered issues, or making changes to existing features to meet client, technology, and user needs.

Understanding what your client's and team's involvement will be after the initial release of your project is important so you can manage and prepare for this during the initial phases. Knowing that your client expects regular maintenance after launch will help guide your team in making critical design and development decisions up front so that you can enable faster and more reliable updates later.

WHO'S WHO IN THE MAINTENANCE PHASE

Generally, the maintenance phase can require your entire team or a subset of your team for new feature development, bug fixing, or other issues that may arise during the life span of the application.

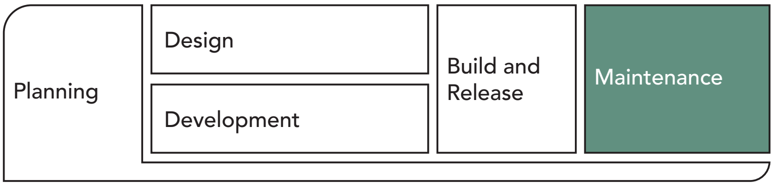

Figure 4-6
The maintenance phase

Depending on the maintenance requirements and needs, you may need to dedicate your entire team for multiple release iterations. Each release may require its own planning, design, development, and build and release phases depending on the scope and needs of the update. Underestimating or not planning the maintenance process can easily make a previously successful project an unsuccessful one.

The chapter in Part VI, “Maintenance,” covers this phase in more detail. We will discuss the maintenance process and how to plan for and manage the continued success of your project application until it is no longer required.

Index

3D animations, 86–87
9-slice scaling, 86

A

about this book, xv–xvi
Acrobat.com Web site, 46, 63, 65
Acrobat Professional, 69, 70
action message format (AMF), 52, 111–112, 113
action sequences, 82
ActionScript
  AMF format and, 111
  animations using, 86
  code snippets and, 87–88
  creating projects using, 124
  Document class in, 121
  Flash Professional settings for, 121, 122
  publishing size and, 50
Adobe Buzzword. *See* Buzzword
Adobe Creative Suite 5, xv
Adobe Fireworks. *See* Fireworks CS5
Adobe Flash Builder. *See* Flash Builder 4
Adobe Flash Catalyst. *See* Flash Catalyst CS5
Adobe Flash Player. *See* Flash Player
Adobe Flash Professional. *See* Flash Professional CS5
Adobe ID, 66
Adobe Illustrator. *See* Illustrator CS5
Adobe InDesign. *See* InDesign CS5
Adobe Integrated Runtime (AIR), 53
  See also AIR applications
Adobe Kuler, 65
Adobe Media Encoder, 92, 93
Adobe Photoshop. *See* Photoshop
Adobe Swatch Exchange (ASE) files, 65, 90
Adobe WorkflowLab. *See* WorkflowLab
adoption rate, 177
agile development, 56
AIR applications
  auto-update support, 178
  code signing certificates for, 163, 166–167, 177–178
  data storage for, 161–163
  debug versions of, 159
  deployment of, 53
  end of life for, 182
  installer badges for, 167
  prototyping, 53
  SOL file in, 161
  versioning, 163, 177
AIR Debug Launcher (ADL), 159
AMF protocol, 52, 111–112, 113
animations
  3D, 86–87
  ActionScript, 86
  building, 86–87
  inverse kinematic, 86
  motion path, 86
  shape, 86
  transitions as, 81
annotated wireframes, 41
  See also wireframes
application programming interface (API), 53
application storage directory, 162
applications
  architecting, 104–106
  data-centric, 51–52
  design changes to, 180–181
  end of life for, 182
  integrating components into, 135
  performance benchmarks for, 150–152
  programs for designing, 78–88
approvers, 20
architecture
  application, 104–106
  component, 106–108, 131
  content, 108–109
  information, 67
  interface, 68–70
  software, 17
artboards, 78, 89
Artboards panel (Illustrator), 89
artwork
  creating components from, 80–81
  importing into Flash Professional, 84
ASDoc utility, 117
ASE files, 65, 90

asset location/management, 144
AsUnit framework, 129
audio design, 16
automated builds, 147
automated testing, 145, 149–150

B

backward compatibility, 176
Badger tool, 167–168
banner ads, 9
benchmarks, 150–152
beta software, 34
bitmaps, 97
BlazeDS, 114
blogs by authors, xvi
brainstorming, 36
branches, 181–182
browser caching, 159–160
budgetary constraints, 33
bug fixes, 179–180
bug scrub process, 157–158
build and release phase, 24–25
build and release process, 24–25, 141–171
  AIR applications and, 161–163
  browser caching and, 159–160
  build process definition in, 146–150
  code signing builds in, 163, 166–167
  debug vs. release builds in, 159
  deployment determination in, 142–143
  environment definition in, 142–146
  Flash Platform features and, 159–163
  local shared objects and, 161
  performance benchmarks in, 150–152
  quality assurance standards in, 147–149
  review process following, 169–171
  runtime shared libraries and, 160–161
  structuring environments in, 143–146
  summary checklists for, 152, 164, 171
  testing in, 147–150, 154–159, 168–169
  verifying projects in, 168–169
build and release teams, 19
build automation tools, 147
builds
  access to, 143
  debug vs. release, 159
  definition of, 142
business plans, 31
business problems, 30–31
buttons
  skin parts, 132, 133
  states, 80, 81, 107, 132
Buzzword, 10
  creative brief creation, 63, 64
  feature prioritization, 37
  specification creation, 41–42

C

caching process, 159–160
CAPTCHAs, 36
certificate authorities (CAs), 166, 167
checklists. *See* summary checklists
classes, naming, 116
clients
  customers vs., 71
  defining for projects, 30
  needs and constraints of, 30
  role of, 14
  test environment for, 146
client-side development, 17
cocktail napkin visions, 31
code coverage, 129
code delivery, 146
code repositories, 144, 182
code signing certificates (CSC), 163, 166–167, 177–178
code snippets, 87–88
coding standards, 116–118
  commenting, 117–118
  naming conventions, 116
  package organization, 117
coding/scripting projects, 17
  ASDoc utility for, 117
  See also development process
ColdFusion, 114
collaboration method, 19–20
color, exploration of, 65
comments
  adding to code, 117–118
  design process, 66–67
communication formats, 110–114
  action message format, 111–112
  basic formats, 110–111
  communication methods and, 112–113
  server-side solutions, 114
component contracts
  creating, 43–45
  executing, 130–133

components, 44
  architecting, 106–108, 131
  containers vs., 134
  creating from artwork, 80–81, 131–133
  design assets integrated into, 134–135
  integrating into applications, 135
  interactive behaviors on, 82
  iterating on, 135–136
  skins separated from, 105, 107–108
  working with libraries of, 83
computer requirements, 144, 145
constraints, 33–34
containers, 134
content
  architecting, 108–109
  managing, 17–18
  updating, 181
content management system (CMS), 108
continuous testing, 154, 156
contracts
  component, 43–45
  data, 114–116
  designer/developer, 24, 43–45
contributors, 20
Convert to Symbol dialog box (Flash Professional), 85, 86
Cover Flow interface, 16
creative briefs, 23, 63, 64
Creative Suite 5, xv
critical fixes, 179–180
cross-domain files, 168–169
CS Review service, 66, 69
custom folder locations, 162–163
custom frameworks, 105
CVS client, 129

D

DACI model, 19–20
data
  adding to designs, 82–83
  caching of, 160
data contracts, 114–116
data layer, 109
data objects (DOs), 111
Data Services panel (Flash Builder), 115
database administrators (DBAs), 17
data-centric applications, 51–52
debug builds, 159
decision paths, 75–76
dedicated process/team, 147
dependencies, 36–37
deploying finished projects, 165–171
  changes made after, 180–182
  code signing and, 166–167
  fixing issues after, 179–180
  installing an AIR badge, 167
  maintenance process after, 175–183
  review process after, 169–171
  verifying and, 168–169
deployment environment
  build and release process and, 142–143, 146
  planning process and, 52–54
deserializing data, 110
design comps, 71
design patterns, 104
design phase, 23
design process, 23, 61–100
  adding details in, 74–78
  emotion and intent of, 62–64
  illustration of layers in, 74
  mapping the user flow in, 74–76
  post-deployment design changes, 180–181
  preparing for the handoff in, 94–99
  structuring creativity in, 67–71
  style guide used in, 65–67
  summary checklists for, 72, 100
  syncing with your team in, 71, 99
  visual design in, 78–94
  wireframe creation in, 76–78
design roles, 14–16
designer and developer contract, 24
  creating in planning process, 43–45
  executing in development process, 130–133
Design-Time Data (DTD) panel, 83
desktop deployment, 53
detailed wireframes, 71
development environment, 143–145
development phase, 24
development process, 24, 103–137
  application architecture in, 104–106
  component architecture in, 106–108
  content architecture in, 108–109
  data contract development in, 114–116
  defining project structure in, 120–126
  distributing project to team members in, 129
  establishing coding standards for, 116–118
  executing designer and developer contract in, 130–133

development process (*continued*)
- expanding project structure in, 126–128
- handing off projects for, 97–99
- initial project development, 120–129
- integrating design assets in, 134–135
- server-side architecture in, 109–118
- summary checklists for, 118, 137
- supporting tiers in, 134
- syncing with your team in, 118, 136
- testing process in, 128–129

development roles, 17–18
diagrams
- architecting applications through, 104
- creating for workflows, 74, 75

discipline map, 14, 15, 18
disconnected handoff process, 43
distribution process, 25
- build process and, 142–143
- development team and, 129

Document class, 121
documentation
- of emotions, 63, 64
- or environment setup, 145

driver role, 20
duplicate items, 91

E

E4X format, 111
edge cases, 148
EightShapes Unify kit, 68
embedded fonts, 92, 93, 94
emotions
- documenting, 63, 64
- identifying, 62–63
- picturing, 63

encrypted local store, 162
end of life, 182
environments
- deployment, 146
- determining, 142–143
- development, 143–145
- documentation for, 145
- reason for multiple, 146
- structuring, 143–146
- testing, 145–146

experience constraints, 34
expired certificates, 178
exporting
- artwork to Flash Professional, 84, 85
- graphics from Flash Professional, 92

external dependencies, 11
external roles, 14
Extreme Programming (XP), 22

F

failed projects, 4
feature sets, 35–37
- brainstorming, 36
- refining, 54

features
- initial estimates of, 37
- post-deployment, 181
- prioritizing, 36–37
- specifications for, 40–42
- testing, 154, 155

feedback, user, 34–35
file size, 50, 150
Fireworks CS5
- using with Flash Catalyst, 92
- wireframe creation in, 68

FLA files, 98, 121
Flash Access, 51
Flash Builder 4
- data-centric applications and, 51–52
- defining projects in, 122–125
- FlexUnit integration in, 41, 129, 149
- integrating design assets in, 134–135
- memory profiler in, 151
- package organization and, 117
- remote services and, 115
- runtime shared libraries and, 161
- unified configuration for, 145

Flash Catalyst CS5
- data-centric applications and, 51
- defining projects in, 125–126
- designing Flex applications using, 78–83
- Fireworks used with, 92
- font embedding in, 92, 94
- handing off projects using, 99
- importing TLF text into, 88
- integrating design assets in, 135
- layers organized in, 80, 96, 97
- organizing projects in, 96–97
- prebuilt components in, 44
- wireframing tools in, 68

Flash Lite 4, 53–54
Flash Media Servers, 51
Flash Platform, xv
- communication formats, 110–114

deployment options, 53
evolution of, 5
Flash Player
backward compatibility of, 176
FPS setting and, 150
Lite version of, 53–54
memory leaks and, 152
penetration and adoption, 176–177
pixel preview mode and, 90
version updates, 176
Flash Professional CS5
ActionScript settings in, 121, 122
building animations in, 86–87
code snippets used in, 87–88
data-centric applications and, 52
defining projects in, 120–122, 124–125
deployment options and, 54, 55
designing Flex applications using, 84–88
exporting graphics from, 92
Flash Builder projects in, 124–125
font embedding in, 92, 93
graphic symbols and MovieClips in, 84–86
handing off projects using, 98
importing artwork into, 84, 85
integrating design assets in, 134
organizing projects in, 94–96
Photoshop import options, 91
publishing size and, 50
unified configuration for, 145
Flex 4 SDK
component architecture, 43, 106–108
Spark architecture, 105, 107, 131
version updates, 178
Flex library projects, 124
Flex project library (FXPL), 135
Flex projects, 52
architecting, 104–106
creating in Flash Builder, 122–124
designing in Flash Catalyst, 78–83
mobile devices and, 53
publishing size and, 50
skins used in, 105, 107–108
FlexUnit 4 suite, 41, 129, 149
fonts, embedding, 92, 93, 94
FPS (frames per second), 150
frameworks, 104–106
project size and, 106
reasons for using, 105
setting up, 126–128
third-party vs. custom, 105
FXG files, 91
FXP files, 52, 99
FXPL files, 99, 135

G
garbage collection (GC), 152
gold master candidate (GMC), 166
graphic styles, 89–90
graphic symbols
converting artwork into, 85, 86
importing from Illustrator, 91

H
high-level vision, 30–31
HTML template file, 123
HTTP services, 112

I
IDE (integrated development environment), 129
Illustrator CS5
artboards used in, 89
Flash Catalyst document profile in, 88
Flash Professional type settings in, 88–89
graphic styles in, 89–90
pixel preview mode in, 90
style guide creation in, 65, 66
symbols used in, 91
wireframe creation in, 69, 70
workflow diagram created in, 74, 75
importing
artwork into Flash Professional, 84, 85
Photoshop files into Flash Professional, 91
TLF text into Flash Catalyst, 88
InDesign CS5
Buzzword links and, 41, 42
wireframe creation in, 68–69
information architecture, 67
information wireframes, 68
See also wireframes
informed category, 20
initial estimates, 37
initial project development, 120–129
defining project structure, 120–126
distributing project to team members, 129
expanding project structure, 120–129
testing process and tools, 128–129
installer badges, 167
integration process, 25
integration testing, 154, 155–156

intent of design, 62–67
interaction design, 16, 82
interface. *See* user interface
inverse kinematic animations, 86
issue tracking, 145
iterative development, 56–57

J
JSON (JavaScript Object Notation), 110

K
keyframes, 86
kick-off meeting, 32–33
kuler, 65

L
large projects, 8, 10–11
layers
  importance of naming, 97
  layer comp feature for, 91
  organizing in Flash Catalyst, 80
  simplifying for FXG, 91
layout design, 15–16
libraries
  setting up third-party, 126–128
  working with component, 83
  See also runtime shared libraries
Library panel
  Flash Catalyst, 83, 96, 97
  Flash Professional, 95
libs directory, 122, 123
LiveCycle ES2, 114
load time, 150–151
local shared objects (LSOs), 161
logic files, 43
logos, 66
long-range specifications, 40

M
Mac OS X
  application storage directory location, 162
  SOL file location, 161
maintenance phase, 25–26
maintenance process, 25–26, 175–183
  adding new functionality, 181
  fixing issues after deployment, 179–180
  leveraging source control, 181–182
  making design changes, 180–181
  recognizing end of life, 182
  summary checklist on, 183
  technological impacts on, 176–179
  updating content, 181
major revisions, 176
management, role of, 14
manual testing, 156
Mate Framework, 115
Media Encoder, 92, 93
media-rich products, 51
medium-sized projects, 9–10
meetings
  kick-off, 32
  postmortem, 169–171
memory leaks, 152
memory usage, 151
metadata tags, 131
methods
  collaboration, 19–20
  naming, 116
mind mapping, 36
minimum system requirements, 151–152
minor revisions, 176
mobile device deployment, 53–54
models, 111
modules, 109, 179
mood boards, 63, 64
motion design, 16
Motion Editor panel (Flash Professional), 87
motion path animations, 86
MovieClips, 85–86
multipage in-depth feature overview, 40–41
multiple team access, 144–145
MVC pattern, 104
MXML files, 97, 123, 125
mxmlc compiler, 109

N
naming conventions, 116
Native Process applications, 178

O
Open Source Media Framework (OSMF), 51, 124

P
package organization, 117
perceived time, 151
performance benchmarks, 150–152

performance testing, 154, 157, 169
perpetual beta, 34
Photoshop
  layer comp feature in, 91
  roundtrip extensions for, 81, 91
  simplifying layers for FXG in, 91
Photoshop.com Web site, 63
pixel preview mode, 90
planning phase, 22–23
planning process, 22–23, 29–57
  adjusting your project in, 54–55
  component contract creation in, 43–45
  feature set definition in, 35–37
  high-level vision definition in, 30–31
  project plan development in, 45–47
  scope review/definition in, 33–35
  specification development in, 40–42
  summary checklists for, 37, 48, 57
  task order assignments in, 55–57
  team definition/roles in, 32–33
  tool/technology choices in, 50–54
player penetration, 177
post-deployment issues
  fixing application bugs, 179–180
  making changes/adding features, 180–182
  technology impacts, 176–179
postmortem meeting, 169–171
prebuilt components, 44
prioritization process, 36
project phases, 21–26
  build and release, 24–25
  design, 23
  development, 24
  maintenance, 25–26
  planning, 22–23
project plans, 31, 45–47
  formats for, 45–47
  overview of, 45
project spectrum, 7–11
  large projects, 8, 10–11
  medium projects, 9–10
  small projects, 8–9
project timelines, 45–46
projects
  end of life for, 182
  failure of, 4
  phases of, 21–26
  reviewing, 169–171
  roles for, 13–20
  scope of, 33–35
  specifications for, 40–42
  spectrum of, 7–11
  successful delivery of, 5–6
  three Ts of, 4–5
ProjectSprouts tool, 129
prototypes, AIR, 53
publishing size, 50
push technology, 114

Q
quality assurance (QA), 18–19, 128, 147–149
  developing test cases/plans/suites, 147–148
  writing the tests for, 148–149

R
rasterizing vector objects, 78
redundant objects, 91
refining projects, 54–55
regression testing, 154, 155
release builds, 159
release candidate (RC), 166
release to manufacturer (RTM) build, 166
RemoteObject class, 113
remoting services, 112–113
  Flash Builder 4 and, 115
  non-Adobe, 114
reports, bug, 158
research, user, 34–35
resource modules, 109, 179
reviewing finished projects, 169–171
RobotLegs framework, 121, 123, 127
roles, 13–20
  adjusting, 54–55
  client, 14
  DACI model, 19–20
  defining, 32–33
  design, 14–16
  development, 17–18
  management, 14
  support, 18–19
runtime shared libraries (RSLs), 50, 88, 160–161
  configuring libraries as, 127–128
  Flash Builder 4 and, 161
  unsigned vs. signed, 160
  updating applications for, 178
  versioning, 179

S
schedule constraints, 33–34
scope of projects
    modification of, 54
    review and definition of, 33–35
scripting projects, 17
securing builds, 166–167
security updates, 176
serializing data, 110
server caching, 159, 160
server requirements, 144, 145, 146
server-side architecture, 109–118
    communication formats/methods and, 110–114
    data contract development for, 114–116
    establishing coding standards for, 116–118
server-side communication solutions, 114
server-side development, 17
Service Request Dispatcher (SRD) framework, 115
services layer, 109
shape animations, 86
signed RSLs, 160
signing builds, 163, 166–167
simple feature overview, 40
Simple Object Access Protocol (SOAP), 112
skin class, 131, 132, 134
skin parts/states, 131–133, 134
skinning architecture, 43, 105
small projects, 8–9
smoke testing, 154, 157
SOAP protocol, 112
software architect, 17
SOL files, 161
source control, 181–182
Spark architecture, 105, 107, 131
specifications, 40–42
    Buzzword for creating, 41–42
    exploring types of, 40–41
    team involvement in, 40
spectrum, project, 7–11
SQLite data, 162
src directory, 121
stakeholders, 32
states, 80, 81, 107, 132–133
stress testing, 154, 157, 169
style guide, 65–67
summary checklists
    for build and release process, 152, 164, 171
    for design process, 72, 100
    for development process, 118
    for maintenance process, 183
    for planning process, 37, 48, 57
support roles, 18–19
SWC files, 98, 126
SWF files, 53, 88, 109
SWZ files, 160
symbols
    converting artwork into, 85, 86
    importing from Illustrator, 91
syncing with your team. *See* team sync
system requirements, 151–152

T
Tables application, 46, 47
tagging process, 181
tasks
    assigning the order of, 55–57
    planning the breakdown of, 46, 47
    tracking, 145
team sync, 24
    in design process, 71, 99
    in development process, 118, 136
    in review process, 169–171
teams, 5
    adjusting roles on, 54–55
    defining roles on, 32–33
    experience of, 34
    reviewing projects with, 169–171
    syncing with, 24
technical requirements, 31
technologies, 5
    choosing tools and, 50–54
    impacts on project maintenance, 176–179
templates
    bug note, 158
    HTML, 123
test cases, 147–148
Test Driven Development (TDD), 41, 128, 149
test plans, 148
test suites, 148
testing process, 24, 128–129
    automated testing, 149–150
    bug scrubbing in, 157–158
    defining for projects, 158–159
    developing cases/plans/suites for, 147–148
    executing manual tests, 156
    Flash Platform features and, 159–163

structuring environments for, 145–146
types of tests used in, 154–157
unit testing tools for, 129
verification testing, 168–169
writing tests for, 148–149
text
Flash Professional options for, 87
Illustrator settings for, 88–89
See also type
Text Layout Framework (TLF), 88, 123
third-party frameworks, 105
third-party libraries
setting up, 126–128
updates to, 178–179
See also runtime shared libraries
throw-it-over-the-fence disconnect, 43
time constraints, 33–34
timelines
development of, 45–46
post-mortem review of, 170–171
tools, 5
choosing for projects, 50–54
unit testing, 128–129
visual design, 88–94
transitions, animated, 81
triage process, 158
type
defining guidelines for, 65
Flash Professional tools for, 87
font embedding for, 92, 93, 94
See also text

U
UI. *See* user interface
Unify system, 68
unit testing, 41, 128–129
unsigned RSLs, 160
URL redirects, 182
URL-based calls, 168
use cases, 31, 35–36
user experience (UX), 23
user interface (UI)
architecting, 68–70
breaking into components, 106
testing process, 154, 156–157
user requests, 31
user research, 34–35
user workflows, 74–76

V
value objects (VOs), 111
variables, naming, 116
vector objects, 78, 97, 98
verification testing, 168–169
video design, 16
video-rich products, 51
views, 106
visual design, 15, 78–94
Creative Suite design tools for, 88–94
designing through wireframes and comps, 78–88
visual sketches, 41

W
waterfall workflow, 55–56
Web deployment, 53
Web services, 112
Web site development, 180
Windows OS
application storage directory location, 162
SOL file location, 161
wireframes
annotated, 41
creating linked, 76–78
information, 68
products for creating, 68–70
updating design of, 78–80
visual design using, 78–88
WorkflowLab, 10, 18, 19
project planning with, 45–46
workflows printed from, 55, 56
workflows, 55–57
diagrams of, 74–76
iterative, 56–57
waterfall, 55–56
workspaces, 65
writing tests, 148–149

X
XFL project folder, 95, 96
XML format, 111