ActionScript 3

с шаблони за дизайн

Професионално издание

Джоуи Лот и Дани Патерсън

СОФТПРЕС®

Съдържание

Въведение *13*

ЧАСТ I Успешни проекти **17**

Глава 1 Проектиране на приложения 19

Фаза Анализ 20

Случаи на употреба 21

Как да пишем обстойни случаи на употреба 23

Оформяне на случаите на употреба 24

Използване на UML при анализа 25

Фаза Проектиране 26

Какво представляват CRC-картите 27

Определяне на класовете 28

Определяне на отговорностите на класовете 29

Определяне на сътрудниците 30

Отношенията между класовете в детайли 31

Оформяне на публични API 33

Проектиране с UML 34

Фаза Реализация 35

Конвенции при програмиране 36

Капсулиране 39

Наследяване и композиция 45

Обвързване 51

Тестване 52

Създаване на прости unit-тестове 54

Създаване на асинхронни unit-тестове 59

Обобщение 62

Глава 2 Програмиране с интерфейси 63

Дефиниране на интерфейси 64

Интерфейси, дефинирани чрез интерфейсни конструкции 66

Интерфейси, дефинирани чрез наследяване 68

Кой от начините за дефиниране на интерфейси да използваме 69

Употреба на полиморфизъм 70

Разликата между тип и клас 72

Вземане на решения по време на изпълнение на програмата 73
Обобщение 76

ЧАСТ II Шаблони 77

Глава 3 Шаблонът Модел-Изглед-Контролер 79
Кои са елементите на MVC 81
Модел 81
Изглед 81
Контролер 82
Отношенията между подсистемите 82
Прост пример за MVC 83
Модел ClockData 83
Изглед на аналогов часовник 86
Тестване на аналоговия часовник 89
Изглед на електронен часовник 90
Тестване на цифровия часовник 92
Използване множество изгледи за един и същи модел 93
Извършване на промени в имплементацията на модела 95
Добавяне на контролер 98
Обобщение 102

Глава 4 Шаблонът Сек (Singleton) 103
Създаване на обекти 104
Ограничаване на създаването на нови инстанции 104
Единствена инстанция и глобален достъп до нея 106
Сек или статични членове 108
Изграждане на елементарен Сек 109
Създаване на Сек 109
Извикване на Сек 110
Изграждане на инфраструктура за настройка на приложението 111
Създаване на XML-документ 111
Създаване на класа Settings 112
Извикване на класа Settings 115
Обобщение 116

Глава 5 Шаблоните Метод фабрика (Factory Method) и Шаблонен метод (Template Method) 117
Абстрактни класове 117
Шаблонен метод 118

Метод фабрика 120
Обобщение 124
Глава 6 Шаблонът Пълномощно (Proxy) 125
Виртуално пълномощно 126
Пример за обект, зареждащ изображение 126
Пример за ленива сериализация 128
Дистанционно пълномощно 133
Пример за Flickr-търсене с Пълномощно 133
flash.utils.Proxy 135
Шаблоните Адаптер и Фасада 140
Обобщение 140
Глава 7 Шаблонът Итератор (Iterator) 141
Проблеми при обхождане на данни 141
Елементите на шаблона Итератор 144
Интерфейсът на Итератора 144
Конкретният клас Итератор 145
Интерфейсът на колекцията 147
Конкретният клас колекция 148
Употреба на Итератори 149
Използване на нулеви Итератори 150
Обобщение 151
Глава 8 Шаблонът Композиция (Composite) 153
Интерфейс на елементите 154
Елементите-листа 157
Възловите елементи 158
Създаване на пример за файлова система 159
Обобщение 166
Глава 9 Шаблонът Декоратор (Decorator) 167
Същността на шаблона Декоратор 168
Интерфейс за декориращите/декорираните обекти 169
Конкретен декориран клас 170
Абстрактен декориращ клас 170
Конкретни декориращи класове 171
Декориране на клас-четец 172
Създаване на интерфейс за декориращите/декорираните обекти 173
Дефиниране на абстрактен клас-четец 174
Дефиниране на конкретния декориран клас 175
Създаване на абстрактен декориращ клас 176

Дефиниране на първия конкретен декориращ клас 177
Тестване на декориращия клас 179
Дефиниране на допълнителен конкретен декориращ клас 180
Дефиниране на нов декориран тип 181
Декориране на декоратори 184
Изграждане на визуални и комутативни декоратори 186
Дефиниране на общия интерфейс 187
Дефиниране на конкретните декорирани класове 188
Дефиниране на декориращи класове 189
Добавяне на некомутативни декоратори 191
Обобщение 196
Глава 10 Шаблонът Команда (Command) 197
Същността на шаблона Команда 198
Интерфейсът 198
Конкретните класове за команда и адресат 199
Клиент и инициатор 200
Елементарно приложение с Команда 200
Създаване на класовете Команди 201
Създаване на адресата 203
Създаване на бутон 204
Създаване на контейнери на команди 205
Тестване на приложението 207
Отменяемост на операции и съхраняване на история на последователността им 209
Приложение с отменяемост на операциите 212
Да направим командите обратими 212
Добавяне на извършените команди към историята 215
Добавяне на бутон Undo 216
Приложение с възможност за повторно изпълнение на отменени операции 218
Имплементиране на IRedoableCommand 218
Добавяне на бутон Redo 219
Създаване на игра Proximity с помощта на команди 220
Дефиниране на клас за данните на играчите 221
Дефиниране на клас-колекция, съдържащ играчите 222
Дефиниране на шестоъгълниците 223
Дефиниране на класа за дъската 224
Дефиниране на клас команда за ходовете 228
Дефиниране на клас фабрика 230

Дефиниране на клас за изглед и контрол на шестоъгълниците. 231
Дефиниране на изглед и контрол за дъската 233
Дефиниране на стартиращия клас 234
Обобщение 236

Глава 11 Шаблонът Спомен (Memento) 237
Създаване на отменяеми операции в Proximity с помощта на шаблона Спомен 243
Дефиниране на типа на спомена 243
Създаване на източник на спомена 244
Дефиниране на тип за отменяема команда 245
Актуализиране на основния клас 246
Актуализиране на стартиращия клас 247
Създаване на повторяеми действия в Proximity с помощта на шаблона Спомен 248
Дефиниране на команда за възстановяване на отменени действия 248
Редактиране на класа на фабриката 250
Редактиране на стартиращия клас 250
Обобщение 251

Глава 12 Шаблонът Състояние (State) 253
Пример за прилагане на шаблона Състояние 254
Създаване на класа `SimpleShooter` 254
Създаване на стартиращия клас 255
Проблемите в този пример 257
Капсулиране на състоянията 257
Интерфейсът за състоянията на баскетболния играч 258
Обекти състояния 259
Класът `ShooterStateMachine` 260
Създаване на стартиращия клас 262
Употреба на абстрактни класове 263
Класът `AbstractShooterState` 264
Обекти състояния 266
Класът `ShooterStateMachine` 267
Създаване на стартиращия клас 268
Преходи 269
Преходи, дефинирани в класа `ShooterStateMachine` 270
Преходи, дефинирани в обектите състояния 271
Обобщение 276

ЧАСТ III ActionScript – теми за напреднали 277

Глава 13 Работа със събития 279

Същността на събитията 279
Използване на събития 281
Елементите, от които се състоят събитията 281
Регистриране на слушатели 282
Изваждане на регистрирани слушатели от списъка 284
Фази на събитията 285
Фаза цел (target) 287
Фаза мехурче (bubble) 287
Създаване на диспечери 290
Същността на обектите събития 291
Същността на свойствата target и currentTarget 292
Подразбиращо се поведение 292
Спиране на потока на събитията 292
Изпращане на събития чрез наследяване 293
Интерфейсът `IEventDispatcher` 294
Пример за работа със събития 295
Създаване на класа `IconButton` 295
Създаване на стартиращия клас 297
Обобщение 298

Глава 14 Изпращане и зареждане на данни 299

Зареждане на текст 300
Създаване на класа LimerickData 302
Създаване на класа LimerickView 304
Създаваме на стартиращия клас 305
Изпращане и зареждане на променливи 306
Изпращане на променливи 306
Зареждане на променливи 308
Изпращане и зареждане на XML 311
Изпращане на XML 311
Зареждане на XML 312
Използване на уеб услуги 312
Използване на Flash Remoting 313
Основите на Flash Remoting 314
Извършване на Flash Remoting заявки 315
Оптимизиране на трансфера на данни 318
Кеширане на данните 318
Опашка и пул за заявки 321
Обобщение 322

Глава 15 E4X (XML) 323
Създаване на XML обекти 325
Достъп до свойствата 326
XML филтриране 328
Обхождане на XMLList обекти 329
Пространства от имена 330
Зареждане и изпращане на XML данни 331
Елементарен SOAP пример 331
Създаване на потребителско събитие 333
Създаване на класа, представляващ уеб услугата 334
Създаване на стартиращия клас 336
Обобщение 338
Глава 16 Регулярни изрази 339
Въведение в класа RegExp 340
Работа с регулярни изрази 341
Булева проверка чрез регулярни изрази 341
Търсене на позицията на отговарящия подниз 342
Извличане на съвпадащи поднизове 342
Замяна на поднизове чрез регулярни изрази 343
Флагове при регулярните изрази 344
Флагът Global 345
Флагът Ignore Case 346
Флагът Multiline 346
Флагът Dot Matches Newline 347
Флагът Extended 347
Метазнакове и метапоследователности 348
Знакови класове 350
Указване на количества 351
Използване на групи в регулярни изрази 351
Създаване на приложението Mad Lib с регулярни изрази 355
Създаване на класовете с модела на данните 356
Контроли за въвеждане 361
Създаване на класовете за визуализация 362
Дефиниране на главния клас 364
Обобщение 366

Въведение

С развитието на платформата Flash, размерът и обхвата на Flash приложенията се увеличи. Това развитие доведе до увеличаваща се зрелост на разработчиците. Flash приложенията придобиха нов ръст, а увеличаващото се търсене означаваше, че са необходими повече разработчици, които да развият разбирането си и връзката с кода, който пишат за своите приложения. Шаблоните за дизайн са източник точно за това.

Самият ActionScript е доста прост. Той предполага запаметяване на основния синтаксис, добър справочник за API и прилична развойна среда. Променя се не ActionScript, а архитектурата на приложенията. Истинското предизвикателство е в това, да прецените какви класове да напишете и какво API да използвате за тях. Когато разработвате все повече и повече приложения, ще откриете, че някои неща се повтарят постоянно. Има конкретни начини за постигане на конкретни задачи и за решаването на често срещани проблеми. Тези решения са основите на шаблоните за дизайн. Много от тях вече съществуват. Ако не умеете да ги разпознавате и прилагате внимателно, просто ще пишете приложенията по трудния начин. Може пак да получите добри резултати, но всеки път ще „преоткривате колелото“. Шаблоните за дизайн ви помагат бързо да откривате интелигентни начини за структуриране на приложенията.

Шаблоните за дизайн, разгледани в тази книга, са подмножество от многото шаблони, разработени от програмистите с годините. Избрали сме тези от тях, които смятаме за най-полезни и приложими при разработката на ActionScript приложения. Те – и всичко, което

ще ви кажем за шаблоните за дизайн – нямат за цел да бъдат като Евангелието. Те ще ви служат като ръководство и вдъхновение при разработката на приложения. Разгледаните шаблони са познати на програмистите от години – като ги изучите, ще имате страхотна начална скорост. Все пак е необходимо да разберете, че тези шаблони не са правила. Само защото някой експерт е открил нещо полезно, това не означава, че сте задължени да го използвате. Прочетете всичко с отворено съзнание, но все пак бъдете критични.

Шаблоните за дизайн са изключително разнообразни – от такива, които ви помагат да сте последователни към определен интерфейс, до такива, които задават генерализирани структури. Независимо от това, как работи или изглежда даден шаблон за дизайн, целта винаги е една: да реши даден проблем. Един пример за често срещан проблем е нуждата да създавате обект без да му указвате конкретен тип по време на компилация. Например в графично приложение може да се налага да създавате и визуализирате графика, за която конкретният тип (лентова, линейна и т.н.) да се определя по време на изпълнение. Ако укажете конкретен тип по време на компилация (като например създадете нов BarChart обект), ще бъдете заключени в негъвкава структура, в която типовете се сменят трудно. Тъй като това е често срещан проблем в много приложения, разработчиците са създавали решения много пъти през годините. Тези решения, които са се доказали като най-използваеми, елегантни, гъвкави и прости за имплементация, са изведени като шаблони за начинаещите. Едно решение на проблема с конкретния тип, описан току-що, наричаме шаблона Метод Фабрика (разгледан в глава 5).

Един от положителните странични ефекти от работата с шаблони за дизайн е това, че те ви помагат да създадете общ „речник“ за ефективна комуникация с другите разработчици. Представете си, че нямате в речника си думата „кола“. Всеки път, когато искате да говорите за кола, ще трябва да описвате обекта като „онова нещо с колела, двигател и врати, обикновено за 2 до 5 души, което дава възможност на един от тях да го управлява чрез механизма на волана“. Това определено би затруднило бързата и ефективна дискусия за каквото и да е, свързано с колите. Когато имате дума за нещо, комуникацията е изключително лесна. По тази причина всички шаблони за дизайн си имат имена. Вместо да си говорите за „шаблона, който позволява да прихванете и зададете състоянието на даден обект за постигане на силна капсулация“, можете да кажете просто „шаблона Спомен“.

За тази книга

Заглавието на книгата казва, че тя е за напреднали програмисти на ActionScript 3.0 с шаблони за дизайн. Винаги е малко подвеждащо да наречеш някого *напреднал*. Това, което имаме предвид е, че тя стъпва отвъд стандартното въведение и API интерфейсите, дискутирани нормално във всяка книга за начинаещи. Поради това се налага да направим няколко предположения за вас като читател. Предполагаме, че вече разбирате основните принципи на ActionScript 3.0, включително (но не само) променливи, изрази, обекти, наследяване, както и на базовите класове.

Може да се каже, че тази книга започва от там, където свършват повечето книги по отношение на обектно-ориентирания ActionScript. Въпреки че много книги разглеждат как се създава клас, тази книга ви учи на различни начини за проектиране на класове. Тук ще научите кога, защо и как да използвате композиция и наследяване и как класовете се свързват един с друг, за да оформят даден шаблон.

Книгата е организирана по следния начин:

1. Въведение в принципите на дизайна на приложения
2. Описания на шаблоните за дизайн
3. Разширени теми за напреднали

В книгата представяме множество примери и ви съветваме да ги следвате.

Можете да намерите допълнителна информация и актуализации на официалния уеб сайт за тази книга:

http://www.rightactionscript.com/aas3wdp. Тук ще намерите и библиотеката с класове AAS3WDP, която можете да свалите. Няколко от тези класове се използват в книгата и ще е добре да свалите библиотеката, ако ще следвате примерите в книгата.