PART 1

핵심 Xcode 툴셋

CHAPTER 2 Xcode 워크스페이스 소개
CHAPTER 3 인터페이스 빌더
CHAPTER 4 핵심 아이폰 툴들
CHAPTER 5 Xcode의 프레임워크
CHAPTER 6 Xcode의 모델-뷰-컨트롤러
CHAPTER 7 Xcode 프로젝트 디버깅
CHAPTER 8 Xcode 문서
CHAPTER 9 애플리케이션 개발

CHAPTER 2

Xcode 워크스페이스 소개

애플의 세계적 수준의 개발 툴들이 여러분의 손 안에 있긴 하지만, Xcode는 만만치 않아 보인다. 컴퓨터에 Xcode를 설치할 때가 되었다. 이번 장에서는 Xcode를 받아 (아마도 이미 가지고 있겠지만) 설치하고, Xcode 워크스페이스라고 알려진 코딩 환경에 대해 배우게 될 것이다. 그리고 가장 간단한 애플리케이션을 생성하고 빌드하여 실행할 것이다.

Xcode 개발자 툴들 구하고 설치하기

시작하기 전에 몇 가지 용어들을 설명하는 게 좋겠다. 종종 서로 바꿔서 사용되곤 하지만, Xcode와 Xcode 개발자 툴Xcode Developer Tool은 다르다. Xcode 애플리케이션은 Xcode 개발의 시작점이고, Xcode 워크스페이스에 접근할 수 있게 해준다. Xcode는 코드 개발의 대부분이 이루어지는 곳으로서 여러분의 작업 환경이며, 이번 장의 주제이다. "Xcode 개발자 툴"이라는 용어는 다음 장의 주제인 인터페이스 빌더를 포함해서 인스트루먼트Instruments, 샤크Sharks, 문서 모음들 등의 툴 전체 제품군을 의미한다. 여러분은 이 책의 뒷부분에서 이러한 툴들에 대해 더 많이 배우게 될 것이다.

Xcode 구하기

어쩌면 여러분은 Xcode 개발자 툴들을 이미 가지고 있을지도 모르겠다. 애플은 모든 새로운 맥에 이것들을 포함시키고 있다. 선택적 설치 폴더Optional Installs folder 안에서

Xcode 툴을 발견할 수 있고, `Xcode.mpkg` 파일도 볼 수 있을 것이다. 애플은 그들의 제품으로 얻은 명성에 안주하려 하지 않으며, Xcode도 예외는 아니다. Xcode 기능의 변화 속도를 감안할 때, 여러분이 가지고 있는 Xcode가 이전 버전일 수 있다. 이런 이유로 애플 개발자 모임(Apple Developer Connection – 이후 ADC라고 함) 사이트에 접속하여 최신 버전을 받아오는 게 좋을 것이다.

안타깝게도, 그냥 사이트를 방문해서 다운로드받을 수 없다. 먼저 이 사이트에 개발자로 등록해야 하기 때문이다. 반가운 소식은 이 절차가 쉽고 빠르며 무료라는 것이다. 자, 이 절차를 통과해 보자.

웹 브라우저에 맥 개발자 센터Mac Dev Center 홈페이지 주소인 `http://developer.apple.com/mac/`을 입력하자. 여기서 맥 개발을 위한 수많은 유용한 자료들을 찾을 수 있을 것이다. 하지만 지금은 무료 ADC 온라인 멤버십free ADC Online Membership에 등록하는 링크를 따라 갈 것이다(**[역자 주]** 화면 중간에 'register' 라는 링크가 있다).

ADC 멤버십의 종류는 다양하지만, 여러분의 목적은 Xcode를 다운받는 것이니 종류에 대해서는 나중에 알아보기로 하자. ADC 온라인 멤버십에 등록할 수 있는 버튼을 찾아 클릭한 후, 양식을 완성하자.

그 양식을 완료하면 맥 개발자 센터 홈페이지로 되돌아가게 될 것이다. 로그인 링크를 찾아서 지금 막 생성한 아이디와 패스워드를 사용해 보자. 로그인한 결과는 멤버십 종류에 따라 다르게 보이지만, 최소한 최신 Xcode를 다운받을 수 있는 링크는 있을 것이다. 이 링크를 클릭하면 곧바로 다운로드가 시작될 것이다.

설치 파일이 상당히 크다는 것을 곧 알게 될 것이다. Xcode 3.1.2의 용량이 거의 1기가바이트이다(**[역자 주]** 현재 최신 버전은 3.2.1이며, 크기는 약 750메가바이트 정도이다) 그리고 여러분의 네트워크 연결 속도에 따라 다르겠지만 다운로드하는 데 시간이 조금 걸릴 것이다. 드디어 `xcode312_2621_developerdvd.dmg`(**[역자 주]** 3.2.1 버전은 `xcode321_10m2003_developerdvd.dmg`)라는 이름의 파일을 받게 되었다. 여러분의 맥에 설치하기 위해 더블클릭을 하면 설치 킷이 보일 것이다. 만일 여러분이 최신판 인스톨 DVD를 가지고 설치를 시작하여도 보이게 될 것이다. 즉, 여기서부터는 여러분이 다운받아서 설치하든 인스톨 DVD로 설치하든 똑같다.

Xcode 설치하기

대부분의 맥 소프트웨어처럼 Xcode 설치 방법은 같다. 설치 폴더에서 `Xcode.mpkg` 파일을 찾아서 아이콘을 더블클릭하면 실행된다. 인스톨러가 인증을 위해 라이선스 동의를 묻지만, 설치 과정에서 놀랄 일은 거의 없다. 설치에 대해 애매한 부분은 설치된 파일의 위치와 사용자정의 설치 옵션, 이 두 가지 정도만 있을 것이다. 지금 이것들에 대해 알아보자.

설치 위치

Xcode의 초기 버전에서는 위치를 선택할 수 없었다. 파일들은 메인 디스크 루트의 개발자Developer 폴더인 `/Developer`로 들어갔다. 비록 Xcode 3은 Xcode 설치 위치에 대해 훨씬 더 많은 유연성을 제공하지만, 솔직히 말해서 디폴트는 여전히 `/Developer`이다. 여러분이 다른 곳에 설치할 특별한 이유가 없다면 아마도 최고의 위치일 것이다.

사용자정의 설치 옵션

설치 과정은 여러분을 사용자정의 설치 옵션 화면으로 안내할 것이다(맥 소프트웨어 설치 프로그램들 중에 몇 가지만 이러한 화면을 가지고 있어서 여러분은 거의 볼 수 없을 것이다). 그림 2-1은 이 단계에서 보여주는 것을 나타낸다.

여기의 옵션들은 설치에 대한 제어권을 조금 더 준다. 예를 들어, 여러분이 맥 OS X 10.4 API를 사용하는 애플리케이션을 만들려 한다면, 여러분은 그에 해당하는 옵션을 선택할 수 있다. 여러분이 문서Documentation 옵션을 체크하면, Xcode는 최신 개발자 문서Developer Documentation를 다운로드할 것이다. 이 옵션을 체크하지 않았다는 것은 온라인으로 개발자 문서를 이용하겠다는 것을 의미한다. 여러분은 이 장의 후반부에서 더 자세히 살펴 볼 환경설정Preferences에서 이러한 세팅을 바꿀 수도 있다. 만일 Xcode 개발이 처음이고, 단지 맥 OS X와 아이폰 애플리케이션을 개발하려 한다면, 화면에 있는 그대로 두고 Continue 버튼을 눌러 넘어가자.

모두 잘 되었다면 설치는 문제없이 진행될 것이고, 성공적인 설치The installation was successful라는 메시지를 보게 될 것이다.

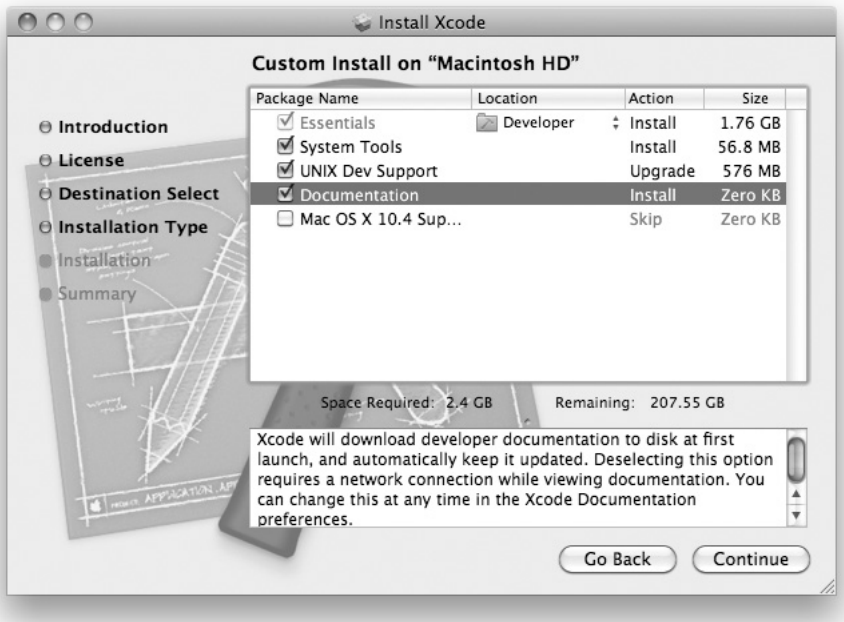

그림 2-1 Xcode 설치 옵션들

설치된 것들

만일 어디에 어떤 것이 설치되었는지에 관심이 없다면 그냥 맘 편히 이번 절을 넘어가도 좋다. 개인적으로, 필자는 컴퓨터에 어떤 소프트웨어가 설치되었는지 늘 관심이 있고 궁금하다. 그래서 설치할 때 어떤 것들이 컴퓨터에 추가되었는지 여기서 살펴보는 것도 좋을 듯싶다.

- `/Developer` 폴더 안에 수많은 애플리케이션, 문서, 그리고 다른 파일들이 있는 하위 폴더들의 최상위 폴더 집합을 가지고 있을 것이다. 여러 이유로 인해 Xcode는 여기에 위치한다.

- `/Library/Developer` 폴더 안에 환경설정 파일과 도움을 주는 파일들이 포함될 것이다. 만일 Xcode를 제거하려는 게 아니라면 이 폴더 안에서 모험하듯 행동하지 말자.

- /usr/bin 안에 Xcode 소프트웨어의 중심에 있는 유닉스 기반의 컴파일러인 gcc 컴파일러와 다양한 지원 파일들이 있다. 여러분은 아마 이것들을 찾으려고 접근하는 일은 절대 없을 것이다.

설치된 Xcode 개발자 툴 제거하기

설치된 Xcode를 제거하기 바란다면 그것은 매우 간단하다. 터미널Terminal을 열고, /Developer/Library 폴더로 바꾼 후, uninstall-devtools 스크립트를 실행한다.

```
$ sudo ./uninstall-devtools -mode=all
```

더 세밀한 조절을 할 수 있도록 이 스크립트에 수많은 다른 옵션들을 줄 수 있다. 이 내용에 대해서는 설치 패키지와 같은 위치에 있는 "About Xcode.pdf" 파일을 참고하자.

여러분은 sudo 명령어 때문에 이 스크립트를 루트root 권한에서 실행해야 한다.

/Developer/Library 폴더에서 제거되었는지 확인해 보자. 만일 제거되지 않았다면 그것들을 휴지통으로 옮기자.

Xcode 워크스페이스 알아보기

만일 Xcode 아이콘이 독Dock에 있다면, 그 아이콘을 클릭하거나 /Developer/Application 폴더에 있는 아이콘(여기에 설치하지 않았는가? 좋다. 그럼, 여러분이 설치한 위치에 있는 아이콘)을 더블클릭하여 다른 맥 애플리케이션을 실행하듯이 실행해 보자.

독Dock에 Xcode 아이콘 놓기

필자는 맥 OS X에서 독Dock의 열렬한 팬이다. 여기에는 마우스 클릭 한 번으로 실행되는 가장 일반적인 애플리케이션들이 위치하게 된다. 또한, 위대한 애플 UI 전문가인 브루스 토그나치니Bruce Tognazzini의 참여로, 화면 가장자리에서 클릭할 수 있는 것들로부터 얻을 수 있는 최고의 사용성을 갖게 되었고, 그 결과 마우스로 독Dock 위의 애플리케이션들을 향해 움직일 때에 지나치는 경우가 거의 없게 되었다.

어찌되었든, 필자는 독Dock에 Xcode 애플리케이션 아이콘을 놓는 것을 추천한다. 이렇게 하기 위해선 먼저 원본 Xcode 애플이케이션을 찾고(설치 시 디폴트로 했다면 /Developer/Application에 있다), 그 아이콘을 독Dock에 드래그하자.

Welcome to Xcode 윈도우

Xcode를 처음 실행할 때 처음 보게 되는 것은 "Welcome to Xcode" 윈도우이다. 이 윈도우는 사실 조금 불쌍하다. 많은 글들과 사용법들은 "만일 여러분이 환영 윈도우를 본다면 'Show' 상자의 체크를 빼라. 다시는 나타나지 않을 것이다."라는 간단한 설명으로 이 윈도우에 대해 일축하곤 한다. 그것은 불행한 일이다. 왜냐하면 사실 이 윈도우는 새로운 Xcode 개발자에게 유용한 정보가 풍부하기 때문이다. 이 윈도우를 취소하거나 최소화하는 건 무척 쉽다. 어쨌든, 만일 여러분이 Xcode를 시작할 때 환영 윈도우를 보이지 않게 설정했는데 마음이 바뀌었다면, Xcode를 기본값으로 재설정할 수 있다(곧 나오는 "모든 것이 끔찍하게 잘못 됐어요." 절 참고).

Xcode의 시작으로 환영 윈도우를 유지해야 하는 좋은 이유 하나는 최근 프로젝트들의 목록을 보여 주며(Xcode 버전 3.2에서), 한 번의 클릭으로 시작할 수 있다는 것이다.

또한, 클릭 한 번으로 새로운 프로젝트를 시작할 수 있다.

"Getting started with Xcode" 링크는 빠른 시작을 위한 Xcode 문서들에게 데려 갈 것이며, 튜토리얼과 기사, 그리고 참고 문서들을 얻을 수 있는 좋은 시작점이 될 것이다. 8장에서는 문서에 대한 주제로 돌아갈 것이다. 하지만, 여러분이 Xcode에서 어떤 것에 대한 이해가 필요할 때, 이곳을 첫 번째 호출 지점으로 하는 게 좋다.

아마 여러분은 Xcode를 빨리 사용하고 싶어할 것이다. 자, 이제 환영 윈도우에서 떠나도록 하자. 지금은 이 윈도우를 최소화하거나 닫겠다. 하지만, 난 여러분이 Xcode의 특징, 툴, 그리고 환경에 대해 정말로 익숙해질 때까지 "Show at launch" 체크박스를 체크해 둘 것을 추천한다.

Xcode의 깨끗한 스크린 뷰

이클립스나 리얼베이직과 같은 IDE[통합개발환경]에 익숙하다면 패널, 인스펙터, 그리고 마법사로 가득 찬 화면을 기대했을 것이다. 환영 윈도우에서 벗어난 후, Xcode 워크스페이스의 첫 인상은 그리 감동스럽지 않을 것이다. 상단의 메뉴바를 제외하곤 바로 볼 수 있는 사용자 인터페이스는 없다. 바로 이것이 Xcode 개발 철학의 한 부분이다. 여러분은 이 책을 통해 Xcode 개발 철학에 대해 계속해서 다시 보게 될 것이다.

일반적으로 개발자들은 개발 시에 자기가 필요한 것만 보려고 한다. 일단 지금은 실제

로 아무것도 하지 않기 때문에 조작할 만한 어떠한 사용자 인터페이스가 없는 것이다.

Xcode 워크스페이스에 대해 알아보기 위해서 소프트웨어 프로젝트를 생성해야 할 것이다. 앞으로의 내용들을 통해 만들고 싶은 다양한 애플리케이션을 위한 프로젝트 타입들을 많이 배우게 될 것이다. 하지만 여기서는 개발환경에 대해 알아보려고 하기 때문에 매우 간단한 애플리케이션을 생성하고 실행할 것이다.

새 프로젝트 시작하기

환영 윈도우가 아직 떠 있다면, 그곳에 있는 링크를 클릭하여 새로운 프로젝트를 시작할 수 있다. 환영 윈도우가 없다면 File ➤ New Project... 메뉴(단축키 Shift⌘N)를 클릭하자. 그림 2-2에서 보듯이 Xcode는 다양한 프로젝트들을 위해 템플릿들을 제공하지만, 대부분의 경우에는 이 템플릿들 중에서 몇 가지만 주로 선택할 것이다.

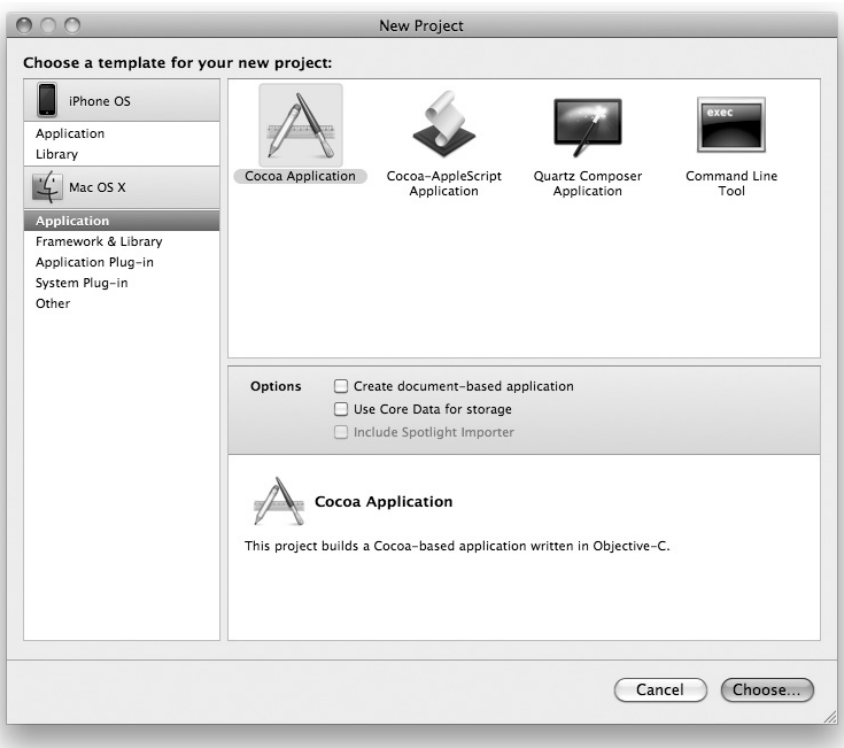

그림 2-2 새 프로젝트 생성하기

NOTE

책을 쓰는 지금, Xcode 툴과 아이폰 툴이 두 개의 분리된 설치 킷으로 제공되고 있다. 만일 지금 Xcode 툴이 설치되어 있다면, 맥 OS X 개발 프로젝트를 위한 옵션들만 보일 것이다. 그러나 맥 OS X와 아이폰 툴이 함께 제공된 것을 설치했다면, 아이폰 개발을 위한 옵션들도 보일 것이다. 아이폰 개발에 대해서는 4장에서 더 많이 알아볼 것이다.

Cocoa Application을 선택하자. Document-based application과 using Core Data 체크박스가 보일 것이다. 이것들에 대해서는 곧 알아보기로 하고, 지금은 체크하지 않고 남겨두자. Choose... 버튼을 클릭한다.

여러분의 새 프로젝트를 위해 이름과 위치를 입력해야 한다. 원하는 이름이나 위치를 지정할 수 있지만, 여기서는 "My First Project"라고 이름을 주고 Documents에 위치하도록 한다. 약간 규칙이 없어 보이겠지만, 이 책의 후반부에 프로젝트들을 어디에 위치해야 하는지에 관한 체계적인 접근법에 대해 배울 것이다. 지금의 프로젝트는 이번 장이 끝나면 사용하지 않을 일회용이기에 여러분이 이름을 무엇이라고 하든지, 어디에 두든지 상관없다. 또한 환영 윈도우에 최근 프로젝트들을 보여주고 있기 때문에 여러분은 이것을 다시 찾을 수 있을 것이다.

마침내 우리는 그림 2-3과 같은 사용자 인터페이스를 갖게 되었다. Xcode 워크스페이스에 온 것을 환영한다.

여기에 엄청난 일들이 많이 일어났다. 그리고 사실, 아직은 알 필요없는 몇 가지 복잡한 개발환경들이 있지만, 여러분은 위대한 소프트웨어를 개발하고 싶어서 이 책을 읽고 있기 때문에, Xcode의 대단한 기능인 "실행"에 대해 지금 봐야 할 것이다. 툴바를 한번 보자. 그러면 "Build and Run"이라는 이름의 큰 녹색 버튼이 보일 것이다.

그렇다. 여러분은 동작하는 애플리케이션을 보게 되었다. 실제로 흥미로운 것이 아직 없다는 걸 인정한다. 하지만 사이즈가 조절되는 윈도우와 About 상자, 그리고 개발에 필요한 메인 메뉴가 있다(그림 2-4 참조).

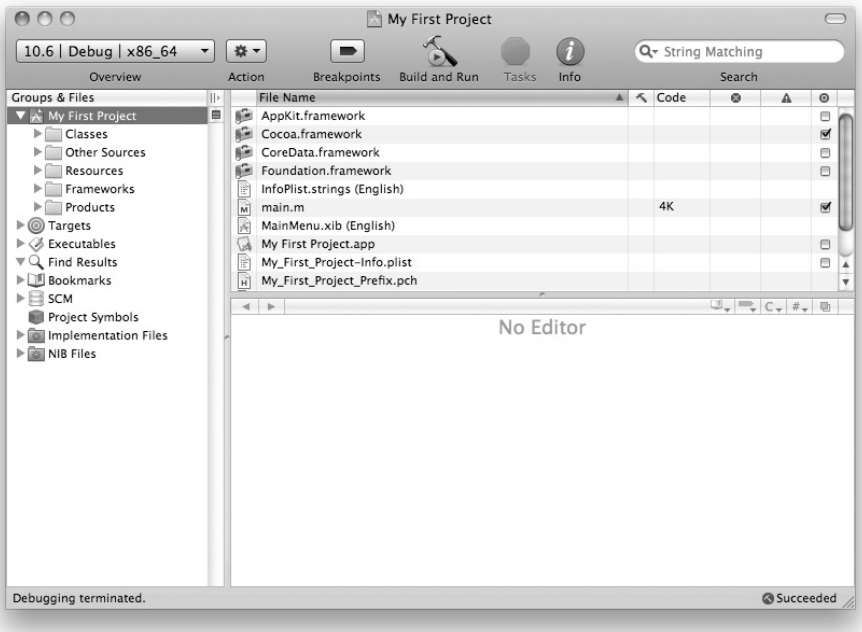

그림 2-3 Xcode 워크스페이스 사용자 인터페이스에 온 것을 환영한다

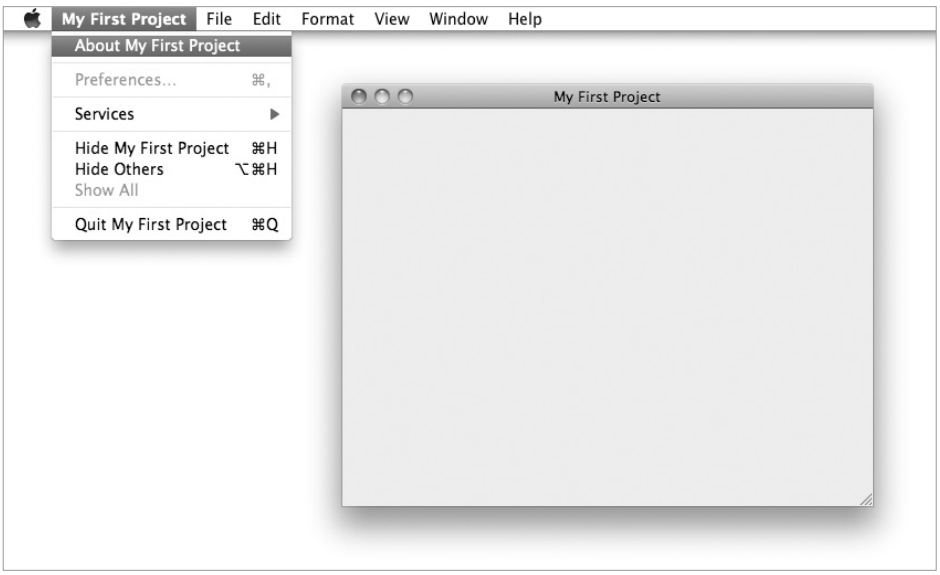

그림 2-4 Xcode 프로젝트의 컴파일과 실행

애플리케이션이 컴파일되고 실행될 때, 몇 가지 유용한 정보들을 얻을 수 있다는 것에 주목하자. Xcode 프로젝트 워크스페이스의 하단에 있는 상태 바는 현재 컴파일 작업에 대해 표시하며, 디버거Debugger 메시지는 프로그램이 실제로 실행되고 있다는 것을 "GDB: Running..."이라고 확인해 준다. 오른쪽에 있는 상태 바는 진행률을 표시하고 (이 애플리케이션은 전체 프로세스가 몇 초 정도 걸리지만, 프로젝트가 복잡해질수록 시간이 더 길어질 수 있다), 프로젝트가 성공적으로 컴파일되었는지 알려준다.

이 애플리케이션에 대해 더 볼 것이 없다. 종료하도록 하자. 이 애플리케이션은 이미 완전한 맥 OS X 애플리케이션이기 때문에 File ➤ Quit 또는 ⌘Q를 사용하여 종료할 수 있다. 그러나 다른 방법도 있다. Xcode의 디버그 모드에서 이 애플리케이션을 빌드했기 때문에 Xcode에 있는 큰 빨간색 Tasks 버튼을 눌러서 종료할 수 있다. 지금 이것을 눌러 보자. 상태 바에 "Debugging terminated."라는 메시지를 보게 될 것이다.

이제 프로젝트 워크스페이스를 더 자세히 볼 시간이다. 여러분이 지금 당장 알고싶어 하는 흥미로운 주요 항목들과 기능들을 알아보자.

Xcode 워크스페이스 작업 환경

프로젝트 윈도우 Project Window라고 불리는 Xcode 워크스페이스의 기본 레이아웃은 툴바가 있는 단순한 세 개의 패널 화면을 가지고 있다. 만일 애플 메일 또는 아이튠즈를 사용해 봤다면 이 인터페이스가 친숙할 것이다. 필자는 이 인터페이스의 주요 부분들을 간단하게 설명할 것이지만, 나중에 이들 중 몇 가지는 더 깊이 다루도록 하겠다.

Xcode는 All-In-One(이름에서 알 수 있듯이 하나의 윈도우에 모든 컴포넌트들이 있는 레이아웃)부터 분리된 윈도우에 주요 기능들의 일부만 있는 Condensed 레이아웃까지 다양한 종류의 레이아웃을 제공한다. 여러분이 원하는 레이아웃을 선택할 수 있다. 이제 막 프로젝트를 생성했기 때문에 지금 프로젝트를 닫고 레이아웃에 대해 알아보는 것은 조금 무례해 보이니, 지금은 레이아웃에 관한 내용을 잠시 놓아 두자(만일 여러분이 지금 당장 더 알고 싶다면 이 장의 후반부에 "Xcode 워크스페이스 환경설정"이 있다). 항상 그렇듯이, 각 레이아웃은 그만의 장점들을 가지고 있고 개인적인 취향에 따라 선택된다. 여러분이 좋아하는 것을 정했다면 Windows ➤ Default... 메뉴를 선택하여 디폴트로 할 수 있다.

Groups & Files 목록

Groups & Files 목록은 소프트웨어 프로젝트의 모든 컴포넌트들과 자원들의 전체 리스트를 표시한다. 이 목록의 상단에는 프로젝트를 위한 클래스와 기타 컴포넌트들의 편리한 저장 위치를 제공하는 **프로젝트 구조**Project Structure 또는 **소스 그룹**source groups이라고 불리는 것을 포함한 프로젝트 명이 있다. 이것에 제한이 없으니 여러분의 개발 스타일에 맞도록 그룹들을 자유롭게 생성하고, 그 그룹들 간의 소스 파일들을 옮길 수 있다. 예를 들어, 프로젝트에 어떤 클래스 그룹을 만들려 한다면 목록에 있는 프로젝트 명에서 오른쪽 클릭(또는 Control-클릭)을 하고 팝업 메뉴에서 Add ➤ New Group을 선택하면 된다.

여러분이 오브젝티브-C로 개발한다면 소스 그룹은 여러분의 코드인 헤더 파일(`.h`)과 클래스 파일(`.m`)을 포함할 것이고, 소스 파일들을 주로 Classes 폴더와 Other Sources 폴더에 둘 것이다.

소스 그룹 안에는 스마트 그룹smart group들이 있다. 프로젝트를 처음 시작할 때에는 대부분 비어 있지만, 개발이 진행되면서 그 안이 채워질 것이다. 신규 개발자인 여러분은 애플리케이션 개발 관련 정보들을 담고 있는 Targets and Executables 그룹의 내용을 알 필요가 없다. Project Symbols 그룹 역시 지금 당장 알아볼 필요가 없다. 지금 단계에서는 필요 없는 좀 더 어려운 주제이기 때문이다.

표시할 필요가 없는 그룹들을 숨길 수 있다. 목록의 그룹에서 오른쪽 클릭(또는 Control-클릭)을 해서 Preferences를 선택하고, 숨기고 싶은 그룹의 체크를 빼자. 삭제 확인을 묻는 창이 보일 것이다. 걱정하지 말자. 똑 같은 방법으로 다시 그룹을 체크해서 그룹들을 보이게 할 수 있다. 필자는 주로 Targets, Executables, 그리고 Project Symbols 그룹들을 숨긴다. 왜냐하면 이것들은 비교적 초보 개발자인 필자에게 그렇게 의미 있는 정보를 가지고 있지 않기 때문이다.

마지막으로, 여러분의 환경설정에 맞도록 목록 내의 그룹들을 옮길 수 있다. 그룹을 선택해서 목록의 새로운 위치로 드래그해 보자.

처음 시작하는 여러분에게 유용한 것들을 더 자세히 알아보기 위해 골라 봤다.

Find Results

대부분의 최신 맥 소프트웨어처럼 Xcode는 프로젝트가 진행될 때 매우 유용한 스팟라이트 스타일Spotlight-style의 검색 인터페이스를 제공한다. Xcode는 검색어들을 이곳에 저장한다. 그래서 필요할 때 그것들을 찾을 수 있게 한다.

Bookmarks

Xcode는 특정한 위치로 빠르게 돌아갈 수 있도록 북마크를 지원한다. 프로젝트에서 생성한 모든 북마크들은 이곳에 표시된다. 예를 들어, Other Sources 폴더의 `main.m` 파일에 북마크를 만들려면, 흥미를 갖는 부분을 선택하고 오른쪽 클릭(또는 Control-클릭)해서 팝업 메뉴 중 Add to Bookmarks를 선택하면 된다. 또는 Edit ➤ Add to Bookmarks나 ⌘D를 누르자. 북마크를 없애려면 그것을 선택해서 Delete 키를 누르면 된다. 확인 메시지는 여러분이 파일을 삭제한다고 하지만, 걱정하지 말자. 그렇지 않다. 오직 북마크만 지워진다.

SCM

SCM은 Source Code Management의 약자로 버전 관리version control로 더 잘 알려져 있다. Xcode는 프로젝트의 코드에 대한 버전 관리를 위해 강력한 지원 툴을 가지고 있다. 그리고 여러분이 팀 프로젝트를 하든 혹은 혼자 코딩을 하든 좋은 버전 관리는 두말 할 것도 없이 화를 면하게 할 것이다. 그래서 이것을 사용하는 것은 좋은 방법이다. 여러분은 9장에서 Xcode의 SCM 툴에 대해 전부 배울 것이다. 현재 SCM은 비어 있으며, 프로젝트를 소스 관리로 넣을 때까지 그렇게 있을 것이다. 지금부터 소스 관리 관점에서 주목해야 할 프로젝트 컴포넌트들을 보게 될 것이다.

Implementation Files와 NIB Files

프로젝트가 개발되고 더 많은 클래스들과 인터페이스 파일들, 그리고 기타 컴포넌트들을 추가할 때마다 그 각각은 이곳에 나타난다. 이 두 그룹은 다른 그룹들과 다르게 동작한다. 즉, 와일드카드 또는 정규식을 사용하여 이곳에 나타낼 것들을 제어할 수 있다. 예를 들어, Implementation Files 그룹에 `.m` 파일들만 나타나게 하고 싶다면 와일드카드로 `".m"`으로 그렇게 할 수 있다. 이 두 그룹들의 옵션들을 보려면 하나의 그룹을 선택해서 File ➤ Info나 ⌘I를 선택하자.

왜 "NIB"이지?

"NIB"이라는 단어는 공경받을 만한 역사를 가지고 있다. 맥 OS X의 유닉스를 토대로 한 많은 것들은 NeXTStep이라고 불리는 운영체제에서 실행되었던 NeXT 컴퓨터를 위해 개발된 툴들을 따른다. NeXT는 소프트웨어 개발을 위한 종합적인 툴박스를 가지고 있었다. 그리고 Xcode는 이것을 직접적으로 계승했다. 여러분은 Xcode에서 작업을 하면서 NeXTStep을 의미하는 "NS"라고 시작하는 객체들의 많은 레퍼런스들을 보게 될 것이다. 이와 같이 NIB는 NeXTStep Interface Builder의 약자이고, NIB 파일들은 알고 있듯이 Xcode 애플리케이션을 위해 여러분이 만들고 저장한 사용자 인터페이스 컴포넌트 파일이다. 더 복잡하게 하는 것들은, Xcode 버전 3.0 NIB 파일들은 .nib이라는 확장자 대신에 그들의 새로운 XML 기반 구조를 의미하는 .xib라는 확장자를 갖는다는 것이다. 그러나 여전히 NIB 파일이라고 광범위하게 말하고 있다(어쩌면 "XIB"라는 발음이 어렵기 때문일지도 모른다).

상세 뷰

상세 뷰Detail view는 현재 선택된 그룹의 확장을 제공한다. 하나 이상의 그룹을 선택할 수 있으며, 상세 뷰에 그 내용들이 표시될 것이다. 이 뷰는 파일 전체의 자세한 정보를 한눈에 볼 수 있도록 설계되었다. 열(컬럼)은 파일 형식에 대한 아이콘, 파일의 빌드 상태, 에러와 경고들, 버전 관리 상태 등 적절한 정보를 표시한다. 이 뷰의 타이틀 바에서 오른쪽 클릭을 하여 그림 2-5의 팝업 리스트처럼 여러분이 원하는 것을 입맛에 맞게 설정할 수 있다.

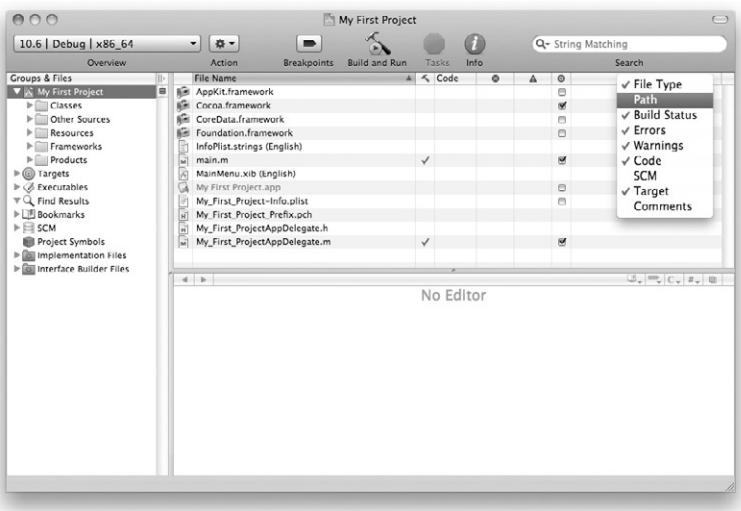

그림 2-5 상세 뷰에서 표시 가능한 정보들

에디터 뷰

에디터 뷰Editor View는 프로젝트에서 에디트할 수 있는 모든 파일들을 가장 빠르게 접근할 수 있도록 해 준다. 이 의미는 Groups and Files 목록 또는 상세 뷰에서 파일을 선택하면 그 내용이 에디터 뷰에 나타난다는 것이다.

Xcode는 코드를 수정할 수 있는 두 가지 방법을 제공한다. 첫 번째는 프로젝트 워크스페이스 윈도우 안에서이고, 두 번째는 분리된 윈도우에서이다. 분리된 윈도우에서 코드를 보려면 상세 뷰의 파일명을 더블클릭하자.

어떠한 에디팅 윈도우를 선택하든지 간에 여기서 보게 될 것과 어떻게 수정할지는 선택한 파일의 형식에 따라 다르다. 어떻게 동작하는지 알기 위해서 Groups and Files 목록에 있는 프로젝트 구조의 최상위 폴더를 선택하자. 여기서 그 이름은 여러분의 프로젝트 명과 같이 "My First Project"일 것이다. 이 폴더의 내용이 상세 뷰에 보일 것이다. 만일 상세 뷰의 첫 번째 항목들(`.framework` 파일들)을 선택했다면, 에디터 뷰에 아무런 변화도 보지 못할 것이다. 왜냐하면 이 파일들은 수정할 수 없기 때문이다. 사실, 이 목록에서 수정할 수 있는 파일들은 `My_First_ProjectAppDelegate.h`, `My_First_ProjectAppDelegate.m`, `Info.plist`, `InfoPlist.Strings (English)`, `Simple application_Prefix.pch`, 그리고 `main.m`이다.

`Info.plist`를 선택하면 이 프로젝트를 위한 plist 파일의 테이블 형식 뷰가 보일 것이다. 만일 프로퍼티 리스트 에디터Property List Editor에서 이 파일을 열었다면, 기본적으로 같은 뷰를 볼 것이다. 하지만 그런 파일들을 접근하기에는 에디터 뷰가 편리하다. 13장에서 프로퍼티 리스트 에디터와 plist 파일들에 대해 더 자세히 배우겠지만, 기본적으로 그 파일들은 키 값을 가지며 바이너리 형식으로 저장된 설정 파일들이다(애플이 이렇게 한 이유를 정확하게는 모르지만, 현재까지 plist 파일들은 간단한 텍스트 기반의 XML 파일이다). 표 형식의 표현은 드롭다운 목록에서 선택하거나 새로운 것을 추가하는 방법으로 기존 행들을 수정할 수 있게 한다.

plist 파일 항목 다음으로 `InfoPlist.Strings (English)`에 대해 알아 볼 것이다. 이 파일을 선택하면 에디터 뷰에 텍스트 파일(주석을 제외하면 현재 비어있는 상태임)이 보인다. 이 파일은 다른 언어들을 사용하는 애플리케이션을 개발할 때 문자열의 지역화 버전을 위해 사용된다. 그렇지만 가장 흥미로운 파일들은 `My_First_ProjectAppDelegate.h`, `My_First_ProjectAppDelegate.m`, 그리고 `main.m`이다. 이것들 중에 하나를 클릭하면 마침내 여러분은 어떤 코드를 볼 수 있을 것이다!

애플리케이션의 시작점 – MAIN.M

main.m 파일은 모든 Xcode 프로젝트의 한 부분으로 생성된다. 이 책은 오브젝티브-C 언어와 코딩 방법에 대해 다루진 않지만, 소프트웨어 개발의 나머지 부분에 대해 설명할 때 이 파일이 어떤 일을 하는지에 대해 간단하게 알아볼 것이다.

만일 C 프로그래밍을 해 봤다면 이 파일의 구조에 대해 놀라지 않을 것이다. 이것은 전형적인 애플리케이션의 시작점인 메인 함수이다. 코코아 헤더 파일들을 임포트하여 간단하게 코코아 애플리케이션 객체를 인스턴트화하고 종료한다. 파일의 이 작은 것으로부터 거대한 여러분의 애플리케이션이 시작한다. 이 책의 후반부에서 여러분은 애플리케이션이 여기서부터 어떻게 진행되는지 정확하게 알게 될 것이다.

대부분의 코드 개발환경들처럼 Xcode는 인식의 용이성을 위해 키워드와 기호를 강조하는 컬러 코드color-coded 편집 기능을 제공한다. 여러분은 Preferences 윈도우를 통해 컬러 코딩에 대해 제어할 수 있다(이번 장의 뒷부분에 "Xcode 워크스페이스 환경설정" 부분을 보자).

에디터 뷰의 상단에는 많은 팝업과 드롭다운 목록을 통해 특정 파일들과 파일의 기능들에 빠르게 접근할 수 있게 하는 얇은 툴바가 있다. 또한 이 툴바를 사용하여 파일들 간을 앞뒤로 전환할 수 있다. 각 `.m` 파일은 헤더 파일인 `.h` 파일을 가지고 있다. 에디터 툴바 우측의 작은 회색-백색 사각형 아이콘을 클릭하면 `.m` 파일과 `.h` 파일을 토글하면서 보여준다(만일 BBEdit를 사용해 봤다면 이 기능이 친숙할 것이다).

또한, 에디터 툴바는 프로젝트를 탐색하는 정말 빠른 방법이라는 것을 증명한다. 만일 어떤 클래스 파일이 커질 때, 간단하게 함수들을 나타내는 팝업을 클릭하면 모든 함수들의 목록이 보일 것이다. 코드에 Pragma 마크를 사용하여 이 팝업 목록에 여러분만의 표시를 추가할 수도 있다. 만일 에디터 뷰에서 클래스 코드 안에 아래와 같은 문자열을 입력하면, 입력한 곳 이하의 모든 메서드들은 팝업 목록에 "My custom methods"라는 항목의 하위로 표시될 것이다.

```
#pragma mark My custom methods
```

필자는 프로젝트들을 정리하는 매우 유용한 방법을 발견했는데, 여러분은 곧 이것에 대한 예제를 볼 것이다.

COPYRIGHT 문구에 대하여

Xcode에서 새로운 프로젝트나 클래스 파일을 생성할 때, 각 파일의 상단에 간단하게 작성자와 copyright 문구가 주석 안에 있는 것을 볼 수 있다. 그러나 그 값을 설정하는 Xcode Preferences 항목이 소용없는 것처럼 보일 것이다.

사실, 이것에 관한 설정은 Xcode 외부인 맥 OS X의 Address Book에서 관리된다. Xcode는 작성자의 이름을 위하여 Address Book에서 "Me"로 불리는 여러분의 정보를 선택할 것이다. Xcode는 오직 성과 이름만을 이 정보에서 선택한다. 만일 여러분의 정보에 회사명이 있다면 그 이름은 copyright 문구에 들어갈 것이다. 만일 회사명이 없다면 copyright 문구는 "Copyright 2010 __MyCompanyName__."이라고 표시될 것이다.

툴바

프로젝트 워크스페이스의 툴바는 많은 공통 기능을 위한 단축키들을 제공한다. 사용 가능한 단축키들은 선택한 레이아웃에 따라 다르다(표준 옵션들에 대한 것들은 그림 2-6을 보자). 그러나 보통의 애플 소프트웨어와 같이 툴바를 오른쪽 클릭을 해서 Customize Toolbar...를 선택하면 여러분이 원하는 기능들만을 갖는 툴바Toolbar를 구성할 수 있다.

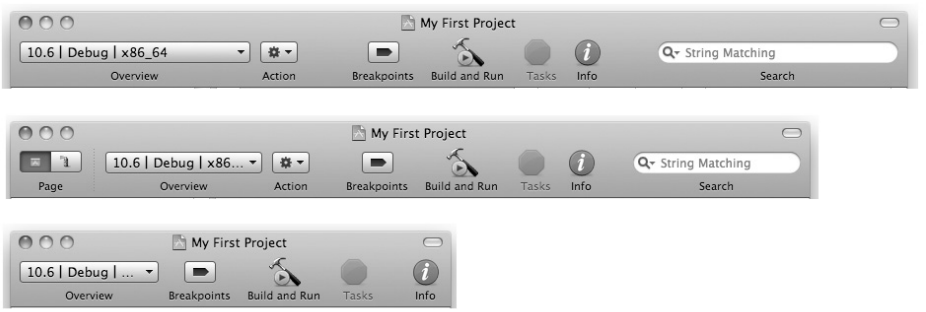

그림 2-6 Defualt, All-In-One, 그리고 Condensed 레이아웃의 툴바 기능들

All-In-One 레이아웃의 툴바에 대해 알아보자. 필자는 이 레이아웃을 좋아한다. 왜냐하면 이것은 한 번 클릭으로 할 수 있는 모든 것들을 가지고 있고, 왼편에 Page 버튼 그룹이 멋지게 표현되어 있기 때문이다. 이 버튼은 애플리케이션이 실행되고 있을 때, 코딩과 디버깅 페이지 뷰 사이를 전환할 수 있게 해 준다.

또 다른 매우 유용한 버튼은 Breakpoints 버튼이다. 이 버튼은 한 번 클릭으로 모든 브레이크 포인트를 켜고 끌 수 있게 해 준다. 여러분이 브레이크 포인트를 설정하면 Build and Run 버튼이 Build and Debug로 바뀌는 것을 알 수 있을 것이다. 만일 Breakpoints 버튼이 비활성화되어 있다면, Build and Debug로 된 모양이 Build and Run으로 다시 바뀌게 될 것이다.

간단한 예를 보이는 게 이해하기 쉬울 것이다. 코드를 추가해 보자. 브레이크 포인트를 하나 추가하고 애플리케이션을 실행해 보자. 만일 여러분이 All-In-One 레이아웃이 아니라면, Xcode는 프로젝트가 열려 있는 동안에 레이아웃 변경을 허용하지 않으므로 여러분의 프로젝트를 먼저 닫자. Xcode ➤ Preferences... 메뉴 또는 단축키 ⌘,를 사용하고, General 설정 패널을 찾아 Layout:에서 All-In-One을 선택하자.

다시 프로젝트를 열고, 상세 뷰에 있는 My_First_ProjectAppDelegate.m 파일을 선택하자. 리스트 2-1은 에디터 뷰에 나타나는 코드를 보여주고 있다([역자 주] 이후 모든 코드는 역자가 직접 테스트한 것이기 때문에 원서와 다를 수 있다. 원서에 오류를 수정할 수도 있고, 주석으로 표현되는 작성자 이름이 저자(Ian Piper) 대신 역자의 이름(Peter)이 표현되기도 할 것이다. 물론, 내용을 변경하지는 않았다.)

리스트 2-1 My_First_ProjectAppDelegate.m 파일

```
//  My_First_ProjectAppDelegate.m
//  My First Project
//
//  Created by Peter on 10. 1. 12..
//  Copyright 2010 __MyCompanyName__. All rights reserved.
//

#import "My_First_ProjectAppDelegate.h"

@implementation My_First_ProjectAppDelegate

@synthesize window;

- (void)applicationDidFinishLaunching:(NSNotification *)aNotification {
  // Insert code here to initialize your application
}

@end
```

앞으로 나올 코드에 대해 벌써부터 걱정하지는 말자. 여러분이 할 일은 단지 콘솔에 로그 메시지를 찍기 위한 코드 한 줄을 추가하는 것뿐이다. 리스트 2-2의 굵은 글씨를 코드에 추가하자.

리스트 2-2 로그 문장 추가하기

```
[…]
- (void)applicationDidFinishLaunching:(NSNotification *)aNotification {
  // Insert code here to initialize your application
  NSLog(@"Application has initialized");
}
[…]
```

TIP

NSLog 명령에는 애플리케이션에 디버깅 메시지를 추가하기 위한 편리한 방법이다. 메시지는 콘솔에 표시될 것이다. 또한 이 명령어는 변수 값을 출력하기 위해 사용할 수 있어서 버그를 찾을 때 유용하다.

이제 브레이크 포인트가 필요하다. 지금 방금 추가한 코드의 왼쪽 회색 부분을 한 번 클릭하자. 그림 2-7과 같이 파란색 화살표를 보게 될 것이다. 이것이 브레이크 포인트이다. 이 지점에 오면 실행을 중단한다.

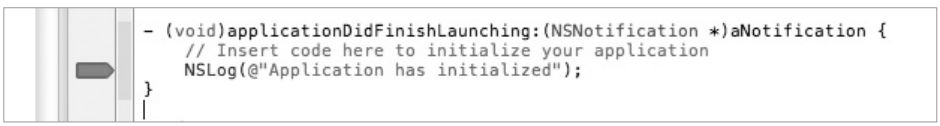

그림 2-7 브레이크 포인트 추가하기

Build and Run 버튼이 지금은 Build and Debug로 변경되었다는 것과 Breakpoints 버튼이 활성화되어 있다는 것을 알 수 있을 것이다. Breakpoints 버튼은 브레이크 포인트를 켜고 끄는 토글 기능을 하게 한다.

좋다. Build and Debug를 클릭하자. 이전과 같이 프로그램이 실행된다. 하지만 브레이크 포인트에서 멈췄다. 상단 좌측에 있는 Page 버튼 그룹에서 Debug 버튼을 클릭하자. 그림 2-8은 실행 중인 애플리케이션을 위한 디버깅 윈도우를 나타낸다.

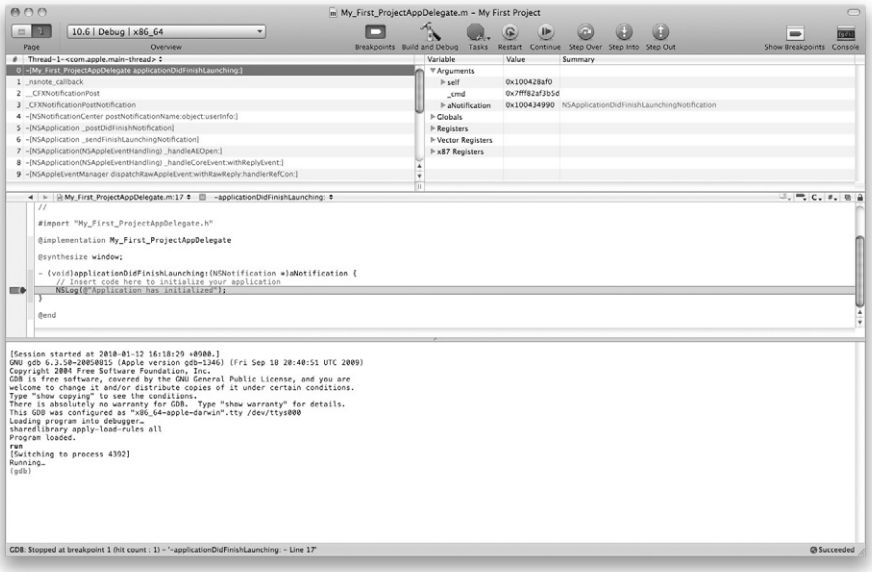

그림 2-8 실행 중인 애플리케이션의 디버깅 윈도우

우리는 7장에서 디버깅에 대해 자세히 알아 볼 것이다. 그러나 지금 여러분은 저 윈도우가 코드 에디터와 현재 변수의 값들을 보여주는 패널, 그리고 콘솔을 가지고 있다는 걸 볼 수 있을 것이다. 콘솔은 gdb[GNU Debugger] 프롬프트를 표시한다. 이제 툴바에 있는 Continue 버튼을 클릭하자. 워크스페이스 뒤에 숨어 있던 애플리케이션 윈도우가 앞으로 나오고, NSLog 메시지가 콘솔에 나타나게 되었다(그림 2-9 참조).

NSLog 메시지를 전략적으로 사용한다면, 콘솔화면 애플리케이션이 실행되는 흐름을 따라갈 수 있는 최고의 장소가 될 수 있다.

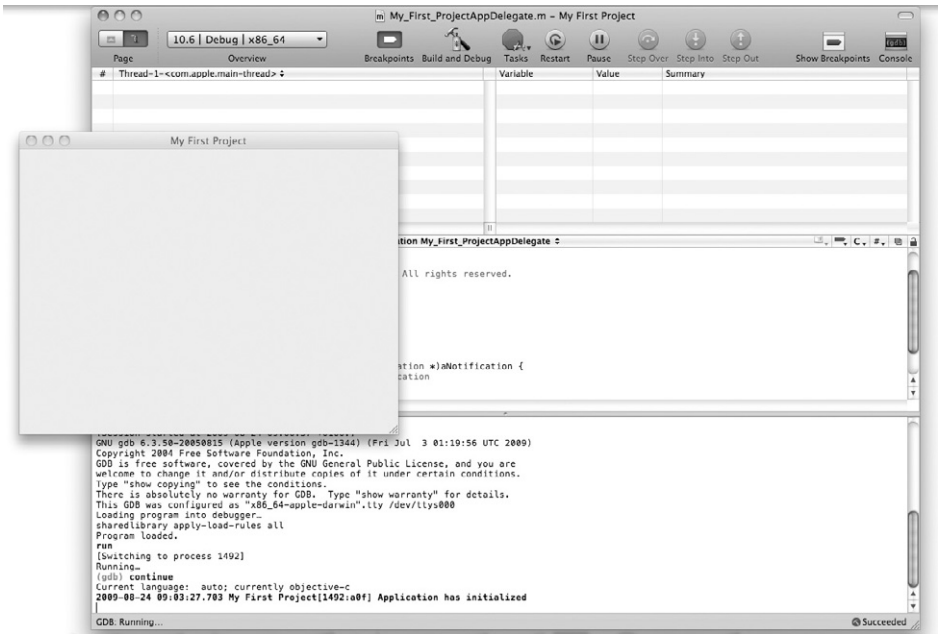

그림 2-9 브레이크 포인트를 지난 애플리케이션

즐겨찾기 바

즐겨찾기 바Favorites Bar는 북마크 바와 약간 유사하다. 그러나 여러분의 작업을 체계화해 주는 또 하나의 방법이다. 원하는 객체를 드래그해서 즐겨찾기 바에 놓으면 된다. 물론, 즐겨찾기 바가 어떤 것인지 먼저 볼 필요가 있다. View ➤ Layout ➤ Show Favorites Bar 메뉴를 선택하자. 즐겨찾기 바에 있는 것을 삭제하려면 그냥 그것을 드래그해서 다른 곳에 놓으면 된다. 필자는 이 기능이 안정적으로 동작한다는 것을 발견했다. 만일 즐겨찾기 바에 있는 것을 바로 위에 있는 툴바로 드래그하면, 그것은 툴바로 추가되는 것이 아니라 그 자리에서 연기를 내며 사라진다.

상태 바

앞에서 Xcode가 여러분에게 액티비티들의 진행현황에 대한 메시지를 보여주는 곳인 상태 바Status Bar를 앞에서 보았다. 이것에 대해 더 크게 다룰 내용이 없다. 만일 상태 바가 유용하다고 생각하지 않는다면, View ➤ Layout ➤ Hide Status Bar 메뉴를 사용하여 숨길 수 있다.

모든 것이 끔직하게 잘못 됐어요

Xcode는 믿을 수 없을 정도로 풍부하면서도 복잡한 개발환경이다. 변경하는 것도 쉽고 실수하기도 쉽다. 다행스럽게도, Xcode 환경은 디폴트로 돌아가기 위한 리셋도 쉽다. 만일 기본 설정으로 복귀해야 한다면 터미널을 실행하여 이 명령어를 입력하자.

```
> defaults delete com.apple.Xcode
> rm -rf ~/Library/Application\ Support/Xcode
```

이 명령어는 Xcode의 plist 파일을 삭제할 것이고, 모든 것을 초기 설정으로 돌려 놓을 것이다. 명심해 두어야 할 것은, 여러분이 이렇게 하면 Xcode의 모든 설정, 디폴트 레이아웃 선택, 파일 히스토리, 툴바 설정 등이 삭제될 것이다. 재미있게도, 북마크는 없어지지 않는다.

Xcode 워크스페이스 환경설정

이 장을 끝내기 위해 Xcode의 Preferences를 둘러보자. Preferences 화면을 보려면 Xcode ➤ Preferences... 메뉴 또는 ⌘,를 누르자.

Preferences 윈도우는 두 부분으로 나뉘어 있다. 상단은 13개 버튼의 스크롤이 달린 툴바이고, 하단에는 이 버튼들에 속한 각각의 상세 화면이 있다. 애플의 다른 소프트웨어들의 환경설정 윈도우들과 꽤 다르다. 또한 설정 사항들에 대한 매우 복잡한 모음이고, 아마도 절대로 변경할 필요가 없는 것들이 많이 있을 것이다. 그래서 우리는 가장 중요한 설정들에 대해서만 다루도록 하겠다.

General

General 버튼은 디폴트로 선택된다. 여기를 선택하는 주요 원인은 툴바 옵션들에서 간단하게 봤던 레이아웃이다. 여기에 세 가지 레이아웃들이 있다.

- **All-In-One.** 이 설정은 하나의 윈도우에 세 가지 패널인 Groups and Files 목록과 상세 뷰, 그리고 에디터 뷰를 보여주는 프로젝트 페이지를 이용하여 모든 요소들을 다 담게 한다. 또한, Project 페이지와 Debugging 페이지 사이를 왔다갔다할 수 있도록 하는 페이지 조절 기능을 툴바에 제공한다. 목록에 있는 파일명을 더블클릭하면 분리된 윈도우로 에디터를 열 수 있다.

- **Condensed.** 이 옵션은 여러분에게 Groups and Files 목록의 변형인 간단한 뷰를 제공한다. 이 목록은 기본적으로 Source Groups을 보여주지만, 트리 구조에서 오른쪽 클릭➤Preferences 팝업 메뉴를 이용하여 다른 항목들을 선택할 수 있다. 다른 레이아웃에서 보여주는 Groups and Files와 달리, Condensed 레이아웃은 항목들의 자세한 정보를 가지고 있다. 바 상단에서 오른쪽 클릭을 하여 변경할 수 있지만, 기본적으로 빌드 상태, 경고, 에러에 관한 정보를 담고 있다. Editor and Debugger 뷰는 분리된 윈도우로 열린다.

- **Default.** 이것은 Editor and Debugger 뷰가 새로운 윈도우에서 열리는 것을 제외하면 All-In-One 레이아웃과 유사하다. 또한, 이 레이아웃은 Groups and File 목록의 우측 상단에 있는 작은 버튼을 더블클릭하여 더 작은 레이아웃으로 축소할 수 있게 해 준다.

Code Sense

여러분이 다른 소프트웨어 개발 툴을 사용해 봤다면, 에디터가 키 입력을 모니터링하고 어떤 것을 입력하려고 하는지 추측하는 **코드 완성**code completion이라는 개념을 가지고 있을 것이다. 이것은 어떤 문구를 완성하기 위해 선택할 수 있는 전체 단어들이나 구문들을 제공한다. Xcode의 코드 센스Code Sence는 지속적인 업데이트를 제공함으로써 더 진보된 단계에 와 있다. 디폴트로, 코드 센스는 켜져 있다. 이 기능을 끌 수 있지만, 보통 코드 작성을 더 빠르게 도와주는 코드 센스를 이용하는 장점들은 솔직하게 단점들보다 훨씬 많다. 그래서 이 기능을 켜 놓은 상태로 두는 게 가장 좋을 것이다. 코드 센스의 정상적인 동작은 여러분이 입력하려는 것을 표시하는 것이다. 이것은 완성하려는 문구를 최적으로 추측하여 회색 문자열로 제공한다. 탭 키를 눌러 그것을 선택할 수 있다. 만약에 제시한 문구가 틀렸을 때, 다음 문자를 입력하거나 Esc 키를 누르면 유사한 다른 항목들이 목록에 나타날 것이다.

이 설정 패널에서 작동을 제어하는 다양한 매개변수들을 선택할 수 있다. 여러분이 수정하고 싶어할 것 같은 것들은 팝업 목록에 메서드의 다양한 매개변수들을 포함할지 말지, 그리고 에디터가 예상된 문구들을 제공받기 전에 기다릴지 말지일 것이다.

Building

이 설정 패널은 컴파일된 코드의 위치를 선택할 수 있게 한다. 디폴트로, 컴파일된 애플리케이션은 프로젝트 폴더에 있는 Build라는 이름의 폴더로 간다. 대부분의 개발자들에게 이곳은 편한 장소이지만, 특정한 다른 곳으로 변경할 수 있다. "For Unsaved Files:" 옵션을 변경하고 싶어할지 모르겠다. 디폴트 값은 "Ask Before Building"이지만, 필자는 "Always Save"로 하고 있다. 또 다른 유용한 옵션은 Message Bubbles 설정이다. 에러가 있는 애플리케이션을 컴파일하면, Xcode는 에디터에서 문제가 되는 코드 위치에 색깔 있는 원을 표시한다. Build Results 윈도우를 띄울지에 대해서도 선택할 수 있다. 기본적으로 Never로 설정되어 있지만, 에러나 경고 시에 Xcode가 그 윈도우를 표시하도록 할 수 있다.

Distributed Builds

이것은 개발자 팀들이나 애플리케이션 개발 작업을 같이 하는 개발자들을 위한 것이다. 이것은 상당히 고급 설정이며, 적어도 처음에는 이곳의 설정들을 바꿀 필요가 없을 것이다. 필자는 지금까지 이 설정들을 바꿔본 적이 없다.

Debugging

이 패널에서 콘솔의 디버거 프롬프트의 모양을 바꿀 수 있지만, 꼭 그래야 할 필요가 없는 것들은 어떠한 변경도 하지 않는 게 좋을 것이다.

Key Buildings

여기에서 Xcode 환경에서의 명령어들을 위한 단축키를 설정할 수 있다. 일반적으로 맥 OS X 소프트웨어의 표준을 따르고 있지만, Xcode를 위한 특별한 단축키도 있다. 만일 예전에 Metrowerks CodeWarrior나 BBEdit와 같은 다른 코딩 환경을 사용해 봤다면, 여기에 있는 Key Binding을 이용하여 그 단축키를 선택할 수 있다. 또 다른 방법으로, 현재의 세팅을 복사하고 원하는 대로 수정하여 여러분만의 단축키를 만들 수 있다.

Text Editing, Fonts & Colors, 그리고 Indentation

이 설정들은 에디터가 어떻게 동작할지를 결정한다. 옵션들은 매우 직관적이어서 더 이상의 다른 설명이 필요 없다.

File Types

이 설정 패널은 Xcode 소프트웨어 개발 중에 작업하게 될 모든 파일 형식들을 나열하며, 이 파일 형식들을 어떻게 다룰지 정의할 수 있게 해 준다. 예를 들면, 애플리케이션에서 사용될 이미지 파일들을 포토샵에서 열 수 있게 하고, PHP 파일들은 TextMate와 연결되게 할 수 있다. 이 패널에 대한 필자의 조언은 다음과 같다. 여러분에게 파일 형식들과의 연결을 바꿔야 하는 매우 좋은 이유가 없다면 디폴트로 설정된 연결을 그냥 놓아두라는 것이다.

Source Trees

이것은 특별한 소스 파일들이 프로젝트 폴더가 아닌 다른 위치에 둬야 하는 개발팀들을 위한 것이다(예를 들어, 개발팀 내의 개발자들끼리 공유되어야 하는 개발자가 직접 만든 오브젝티브-C 클래스들). Xcode에서의 개발하려는 신규 개발자인 여러분은 여기서 바꿀 것이 없을 것이다.

SCM

SCM 환경설정 패널은 소스 코드 관리 또는 버전 관리 시스템 접근에 대한 설정을 하는 곳이다. 이 책의 후반부에서 서브버전Subversion을 설치하는 방법과 이를 사용하기 위해 프로젝트 설정을 어떻게 하는지 배울 것이다. 지금은 여기서 모든 일이 일어난다는 것만 알고 넘어가자.

Documentation

이 환경설정 패널은 개발자 문서를 관리하는 방법을 지정할 수 있게 해 준다. Xcode를 설치할 때, 이것이 옵션이었다는 것을 기억하고 있을지 모르겠다. 체크박스를 선택하면 최신 문서를 항상 유지할 수 있도록 해 주며, 선택하지 않으면 필요할 때마다 받아야 한다. 디폴트로, Mac OS X Snow Leopard Core Library와 Xcode 3.2

Developer Tool Library를 가지고 있지만, 다른 라이브러리들 또한 얻을 수 있다. 그림 2-10을 보자.

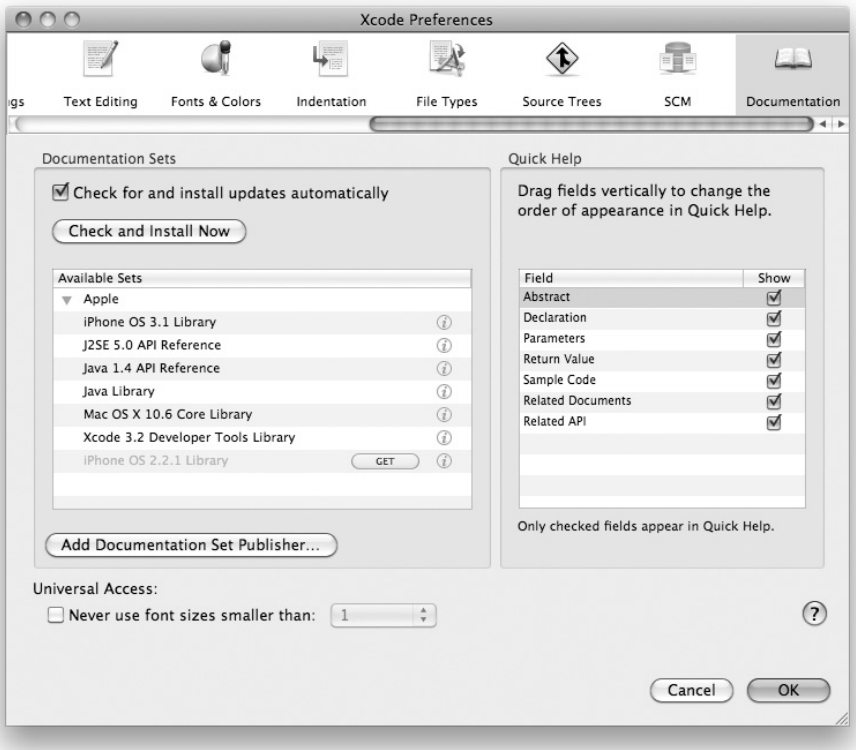

그림 2-10 Documentation 환경설정 패널

요약

그렇다. 지금쯤이면 Xcode 워크스페이스를 어떻게 사용하는지 그리고 원하는 모양을 어떻게 만드는지에 대한 좋은 그림을 갖게 되었을 것이다. 여러분은 강력한 코드 개발 툴들에 대한 감각이 점점 늘게 될 것이다. 그러나 뭔가 빠졌다는 것도 아마 알고 있을 것이다. 맥 소프트웨어의 큰 부분은 굉장한 사용자 인터페이스User Interface, UI이다. 이것은 여러분을 애플리케이션으로 끌어당기며, 사용자 경험User eXperience, UX에 강력한 초점을 두고 있는 매력적인 비주얼 설계이다. 애플 제품인데 Xcode 툴이 사용

자 경험을 창조하기 위한 굉장한 지원 툴들을 가지고 있지 않다고 생각하는가? 당연히 여러분은 기대하고 있을 것이며 결코 실망하지 않을 것이다. 인터페이스 빌더는 맥과 아이폰 전문 개발자들이 선택한 도구이다.

여러분이 다음 장으로 가는 이유가 바로 이것이다.