11. Arbeiten mit dem Organizer

Aus dem Organizer ist mittlerweile ein sehr flexibles Tool für die Entwicklung, Wartung und Distribution von Anwendungen geworden. Es basiert auf der grundlegenden Idee, tägliche Abläufe, die ansonsten mit Xcode und dem Finder ein stetiges Wechseln von Fenstern und Anwendungen erfordern, in nur einem Fenster zu organisieren.

Die Bandbreite reicht von der Projekt- und Sourceverwaltung über den Zugriff auf Entwicklungsgeräte und damit verbundene Möglichkeiten des Debuggens und der Fehlersuche bis hin zur Verwaltung von Provisioning Profiles und Distributionen. Die folgenden Abschnitte sollen Ihnen ein Einstiegspunkt und Ideengeber zum Einsatz dieses mächtigen Werkzeugs sein.

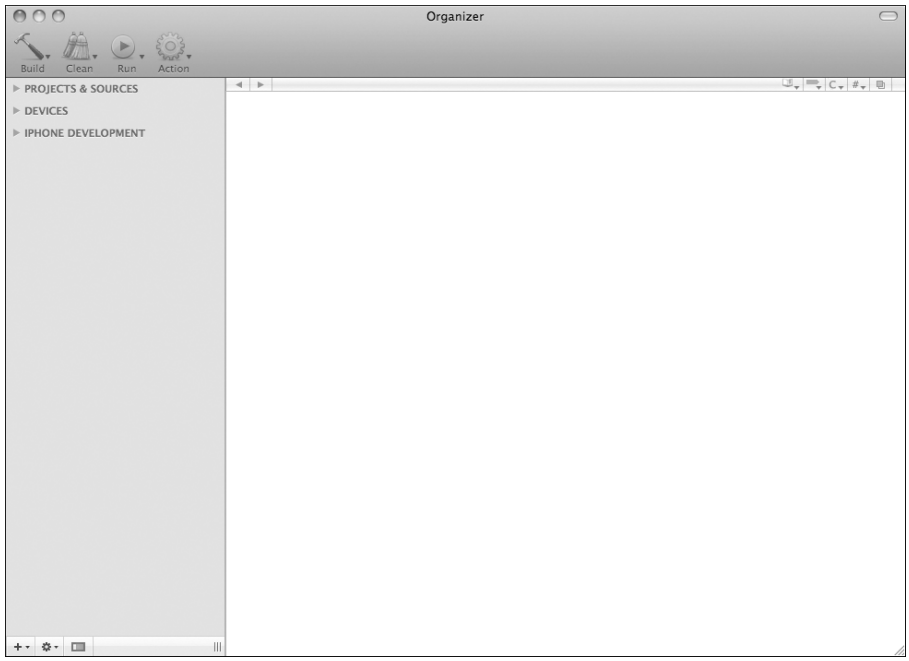
11.1 Das Organizer-Fenster

Der Organizer ist ein Bestandteil von Xcode. Sie starten ihn aus dem Menü *Window/Organizer* oder über die Tastenkombination ctrl+cmd+O. Die Oberfläche nach dem Start ist eher als spartanisch zu bezeichnen, wie die Abbildung unten zeigt. Darin liegt aber gerade der Vorteil: Die Oberflächen sind auf das Wesentliche beschränkt, viele Funktionen sind strukturiert und somit nach ihrer Zugehörigkeit untergebracht. Dennoch verbirgt sich dahinter eine spannende Detailtiefe.

Die Toolbar bietet einen einfachen Zugriff auf Aktionen wie Build, Clean und Run. Im unteren zweigeteilten Bereich ist links eine Navigation mit drei unterschiedlichen Bereichen eingerichtet:

- *PROJECTS & SOURCES*: Unter diesem Bereich lassen sich (ähnlich symbolischer Links) Projektverzeichnisse und andere Ressourcen für einen schnellen Zugriff einhängen. Weitere Informationen dazu finden Sie in Kapitel 11.2.
- *DEVICES*: beherbergt den Bereich der Entwicklungsgeräte. Einmal angeschlossen erhält man als Entwickler Zugriff auf viele wichtige Details. Weitere Informationen dazu finden Sie in Kapitel 11.3.
- *IPHONE DEVELOPMENT*: Dieser sehr generische Bereich versteht sich als Entwicklungsunterstützung. Weitere Informationen dazu finden Sie in Kapitel 11.4.

Neben der Navigation befindet sich ein kontextsensitives Editorfenster. Es zeigt vom Quelltexteditor bis zur Geräteverwaltung Informationen und dient als Eingabewerkzeug.



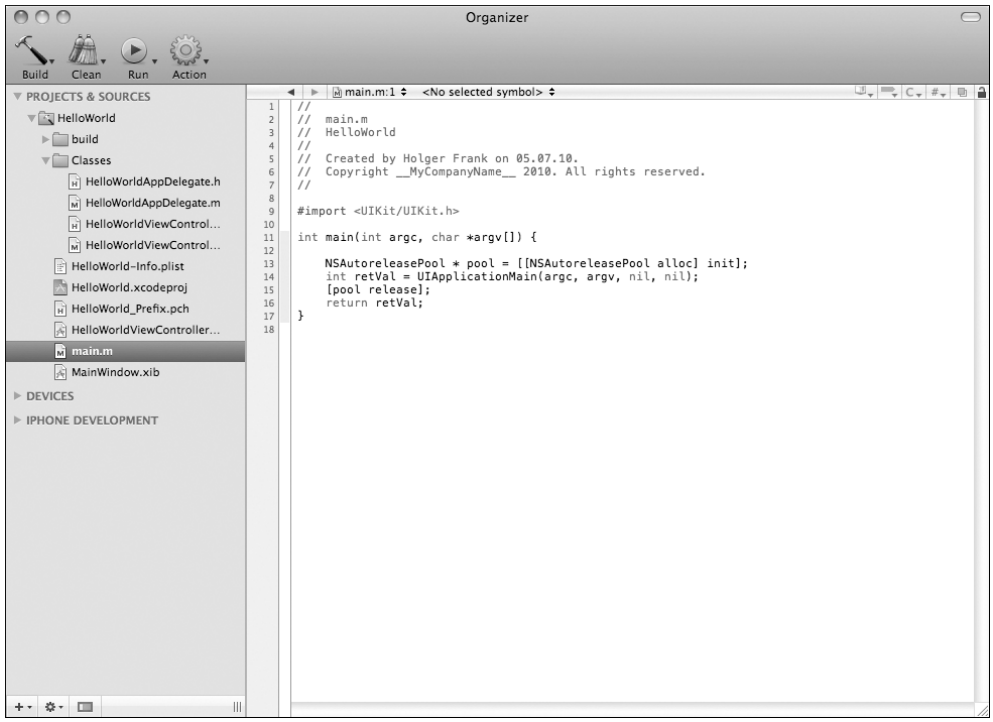
Die auf das Wesentliche reduzierte Oberfläche des Organizers. Unter der Motorhaube finden sich in einem Fenster viele wichtige Funktionen rund um die iPhone-Entwicklung.

11.2 Projekte verwalten mit dem Organizer

Der Bereich *PROJECTS & SOURCES* in der Navigation dient als Schnellzugriff auf Entwicklungsprojekte und beliebige Ressourcen. Dadurch können Sie Ihre Projekte organisieren und die wichtigsten Aufgaben wie einen Build oder den Start der Anwendung ohne das sonst übliche Hin- und Herwechseln zu Xcode erledigen.

Im Folgenden übernehmen wir das Projekt *HelloWorld* aus Kapitel 8 in die Organizer-Oberfläche. Suchen Sie dazu das Projektverzeichnis über den Finder und ziehen Sie den Ordner in den Bereich *PROJECTS & SOURCES*. Alternativ nutzen Sie den Plus-Button unter der Navigation mit der Option *Add Existing Folder* und wählen das Projektverzeichnis aus. Sobald Sie ein Projekt öffnen, aktivieren sich Schaltflächen der Toolbar. Sofern es sich um ein Entwicklungsprojekt handelt, können Sie diese auch einsetzen.

Per Drag & Drop können Sie Verzeichnisse und Dateien in der Ansicht verschieben. Das schlägt sich allerdings auch im Dateisystem nieder – es handelt sich ja wie gesagt im Prinzip um symbolische Links auf Verzeichnisse und Dateien. Unterschiedlichste Projekte und Ressourcen lassen sich so zentral und übersichtlich verwalten. Aktuelle Projekte können Sie in der Ansicht nach oben legen, um noch schneller darauf zuzugreifen, ohne aber die anderen Projekte aus dem Blick zu verlieren.



Die Projektansicht im Organizer. Der Editor rechts ist kontextsensitiv und bietet zu den Standardeditoren von Xcode noch weitere Organizer-Ansichten.

11.3 Zugriff auf Endgeräte

Der zweite Bereich *DEVICES* in der Navigation erschließt Ihnen einen besonderen Zugang zu Entwicklungsgeräten wie das iPhone, den iPod touch oder das iPad. Der Zugriff auf das Gerät ist in vier thematische Register gegliedert:

- *Summary*: stellt eine Übersicht mit Geräteinformationen, den installierten Provisioning Profiles und der Liste aller Anwendungen auf dem Gerät dar und bietet eine einfache Konfiguration.
- *Console*: zeigt die Ausgaben der Konsole auf dem Gerät.
- *Device Logs*: bietet den Zugriff auf die Crashlogs des Gerätes.
- *Screenshots*: erstellt einen Screenshot vom Gerät und überträgt ihn auf den Rechner. Zusätzlich lässt sich ein Screenshot sofort als Startbildschirm verwenden.

Crashlogs auslesen

In der Abbildung unten sehen Sie das Register *Device Logs*. Im oberen Bereich finden Sie eine Liste aller auf dem Gerät gespeicherten Crashlogs – in der Abbil-

dung ist es nur ein einziger. Diese sind gruppiert nach dem Namen der Anwendung, dem Typ des Crashlogs – hauptsächlich *Crash* oder *Low Memory* – und dem Datum sowie der Uhrzeit. Nach jeder dieser Kategorien lässt sich die Liste sortieren. Sobald Sie einen bestimmten Eintrag auswählen, werden im unteren Editorfenster die Informationen zum Crash angezeigt. Diese beinhalten allgemeine Daten zur Anwendung und Laufzeitumgebung sowie ein Stacktrace für jeden Thread. Das Stacktrace besteht aus einer Liste aus Speicheradressen der aufgerufenen Funktionen und Methoden sowie aus der aktuellen Position der Ausführung innerhalb der Methoden. Um mit einem Crashlog wirklich etwas anzufangen, müssen die Speicheradressen wieder in Symbole, Namen und Zeilennummern umgewandelt werden. Das passiert im Organizer automatisch, allerdings muss dafür eine bestimmte Datei des Build-Vorgangs im Suchpfad vorhanden sein. Diese Datei mit der Endung *dsym* sollten Sie für alle Distributionen aufbewahren, damit Sie jederzeit für eine Version im Umlauf einen Crashlog auswerten können. In Kapitel 11.4 wird im Rahmen der Entwicklungsunterstützung eine einfache Möglichkeit der Archivierung beschrieben.

Crashlogs aus Betatests oder App-Store-Distributionen

Crashlogs auf dem eigenen Gerät sind über den Organizer zugänglich. Aber wie kommen Sie an Crashlogs aus Betatests oder einer Anwendung, die über den App Store geladen wurde, wenn kein direkter Zugriff auf das Gerät besteht?

Für App-Store-Distributionen ist die Frage sehr einfach zu beantworten: In iTunes Connect gibt es für jede Anwendung den Punkt *Crash Reports*. Dort werden anonymisierte Fehlerberichte gesammelt und nach den Ursachen gefiltert aufbereitet. Diese werden nach Häufigkeit sortiert angezeigt und können mit einem Klick heruntergeladen werden.

In Betatests müssen Sie Ihre Nutzer bitten, Ihnen die Fehlerberichte per E-Mail oder Dateiupload zur Verfügung zu stellen. Dazu können Sie folgende Schritte an Ihre Tester weitergeben, um an die Crashlogs heranzukommen:

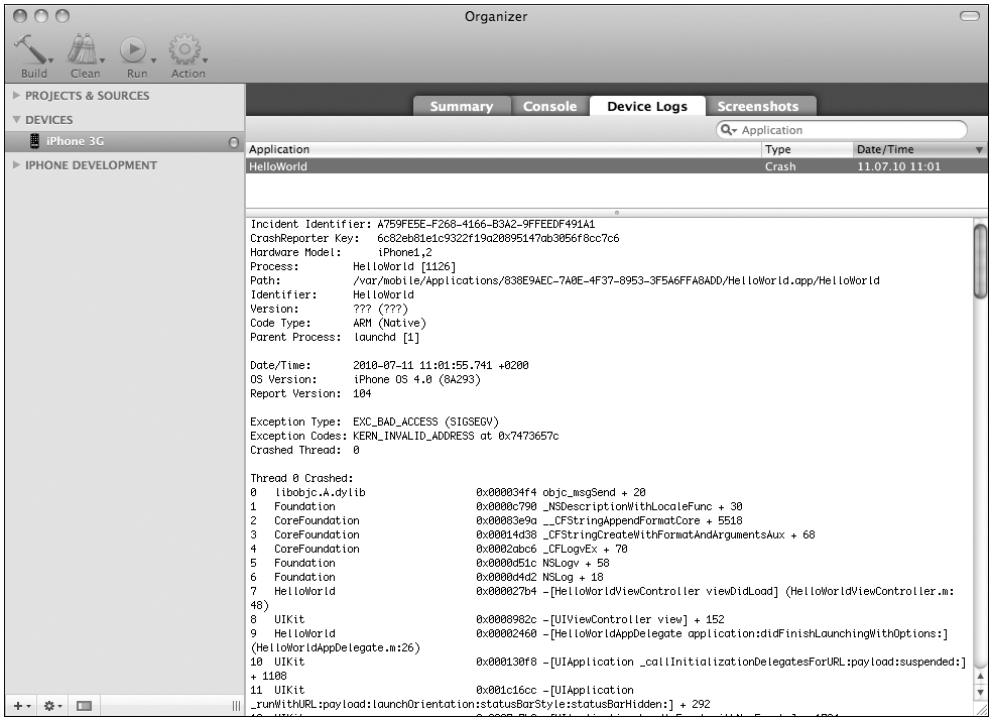
1. Synchronisieren Sie Ihr Gerät mit iTunes.
2. Die Fehlerberichte werden auf dem Rechner abgelegt. Je nach Betriebssystem finden Sie diese in:

   Mac OS: *<user home>/Library/Logs/CrashReporter/MobileDevice/*

   Windows Vista: *C:\Users\<username>\AppData\Roaming\Apple computer\Logs\CrashReporter\MobileDevice*

   Windows XP: *C:\Documents and Settings\<username>\Application Data\Apple computer\Logs\CrashReporter\MobileDevice*

3 An den angegebenen Pfaden finden sich Unterverzeichnisse mit dem Namen des Endgerätes. Diese enthalten die Fehlerberichte, deren Dateinamen mit dem Namen der Anwendung beginnt und einen Zeitstempel enthält. Schicken Sie diese Berichte für die Betaversion einzeln oder gepackt an den Entwickler.



Ein Crashlog im Bereich Devices. Durch die Auflösung von Speicherstellen zu Namen und Zeilennummern lassen sich Ursachen für Abstürze schnell beheben.

Xcode neu starten

Sollten Sie den Eindruck haben, dass die Informationen in den Registern *Console* oder *Crashlogs* nicht aktuell sind, starten Sie Xcode neu. Das ist ein bekannter Fehler des Organizers.

11.4 Entwicklungsunterstützung

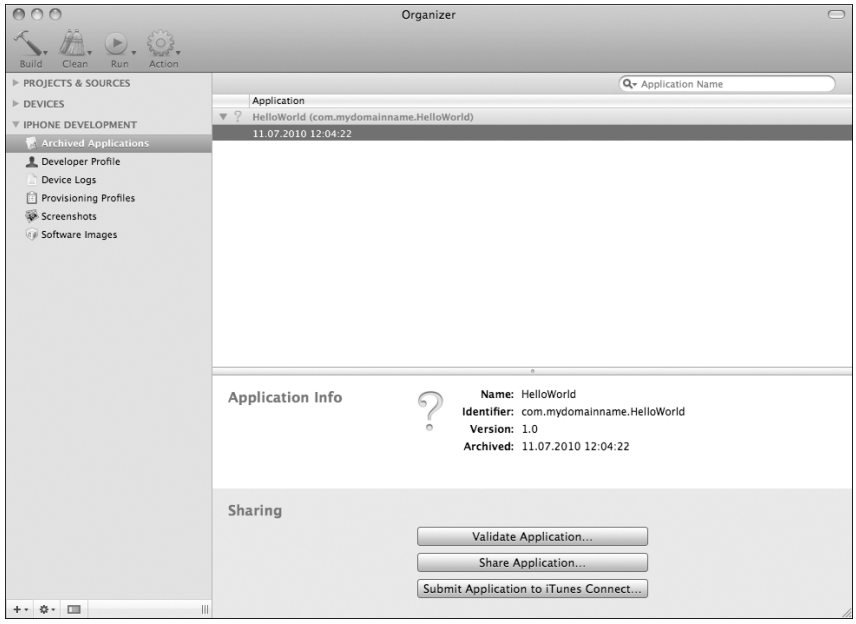
Im Bereich *IPHONE DEVELOPMENT* der Navigation im Organizer ist eine Reihe von Funktionen rund um die Anwendungsentwicklung, Distribution und Archivierung untergebracht. Der Organizer offenbart hier auch eine Schnittstelle zum iPhone Provisioning Portal und zu iTunes Connect.

Wenn Sie also zur Eingabe einer Apple-ID und des Zugangspassworts aufgefordert werden, geht es genau um diese Verbindung. Die folgenden Abschnitte geben einen kurzen Einblick in die verschiedenen Funktionsbereiche.

Archived Applications

Das Anwendungsarchiv hilft Ihnen, Ordnung in Ihre Distributionen zu bringen. Es speichert ein Archiv von Build-Vorgängen jeweils mit Datum, Uhrzeit sowie der zugehörigen Versionsnummer.

Wie kriegen Sie nun eine kompilierte Anwendung in das Archiv? Dafür besitzt Xcode die Option *Build/Build And Archive*. Damit wird wie gewohnt ein Build-Vorgang angestoßen, nur dass am Ende die Anwendung in das Archiv kopiert wird. Zusätzlich wird auch die *dsym*-Datei gespeichert. Das bedeutet, dass Sie sich keine Sorgen mehr um die Auflösung von Speicheradressen bei Crashlogs in eine lesbare Form machen müssen.



Das Anwendungsarchiv hilft, Ordnung in die verschiedenen Versionen Ihrer Anwendungen zu bringen. Dabei wird nicht nur die Anwendung selbst, sondern auch die dsym-Datei zur Auflösung von Speicheradressen in Methodennamen und Zeilennummern archiviert.

Die drei Buttons unter *Sharing* erlauben eine Weiterverarbeitung der Binärdatei:

- *Validate Application*: vergleicht die Anwendung mit vorigen Versionen in iTunes Connect und stellt fest, ob es potenzielle Rückweisungsgründe gibt. Hier sind allerdings nur einige automatische Prüfungen hinterlegt, weswegen ein erfolgreicher Test keine Garantie für eine Freigabe durch das Review-Team ist. Voraussetzung dafür sind ein Account in iTunes Connect, der das

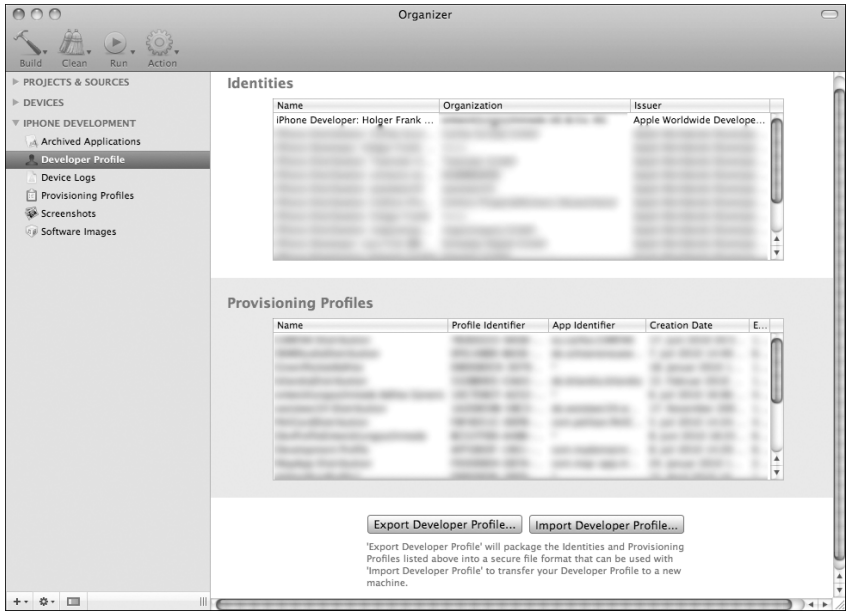
Hochladen von Dateien erlaubt, und ein bereits angelegtes Update mit der richtigen Versionsnummer ohne eine Binärdatei.

- *Share Application*: ermöglicht das Verteilen der Anwendung entweder als Enterprise-Anwendung oder als Betaversion. Dabei kann die Datei zur Distribution entweder auf der Festplatte oder sofort in eine E-Mail kopiert werden. Die Signierung erfolgt mit dem ausgewählten Provisioning Profile.
- *Submit Application to iTunes Connect*: überträgt die Datei zu iTunes Connect. Die Voraussetzungen sind dieselben wie für *Validate Application*.

Gerade die einfache Verteilung als Ad-hoc- oder Betadistribution ist eine deutliche Vereinfachung der Abläufe. Da die beiden anderen Varianten jeweils ein bereits angelegtes Update in iTunes Connect voraussetzen und weniger häufig zum Einsatz kommen, sind es die eher seltener genutzten Möglichkeiten, die dennoch auch eine Erleichterung der Prozesse mit sich bringen. Gerade unter dem Aspekt, dass Apple die Übertragung der Anwendung nun nicht mehr über die Webseite von iTunes Connect akzeptiert, unterstreicht die Wichtigkeit dieser Funktion.

Developer Profile

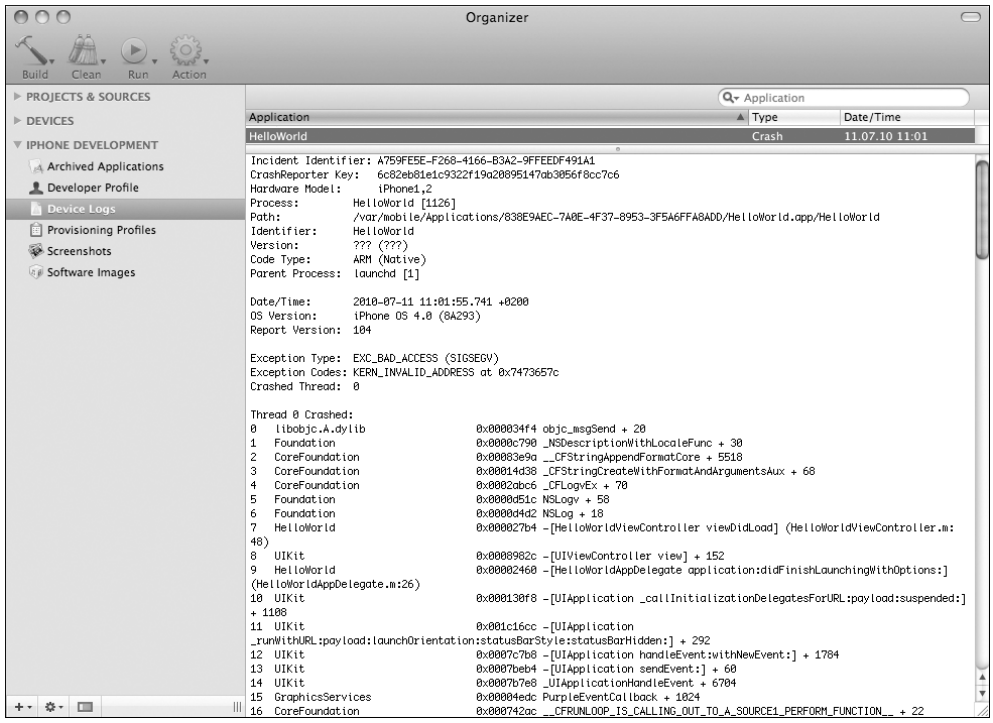
Unter *Developer Profile* können Sie alle Entwicklungsprofile und die zugehörigen Zertifikate und Identitäten importieren und exportieren. Das erzeugte Archiv wird zur Sicherheit mit einem Passwort verschlüsselt. Das vereinfacht die Migration einer Entwicklungsumgebung auf einen neuen Rechner.



Im Developer-Profile-Bereich lassen sich Entwicklungsprofile mit den zugehörigen Zertifikaten und Schlüsseln im- und exportieren.

Device Logs

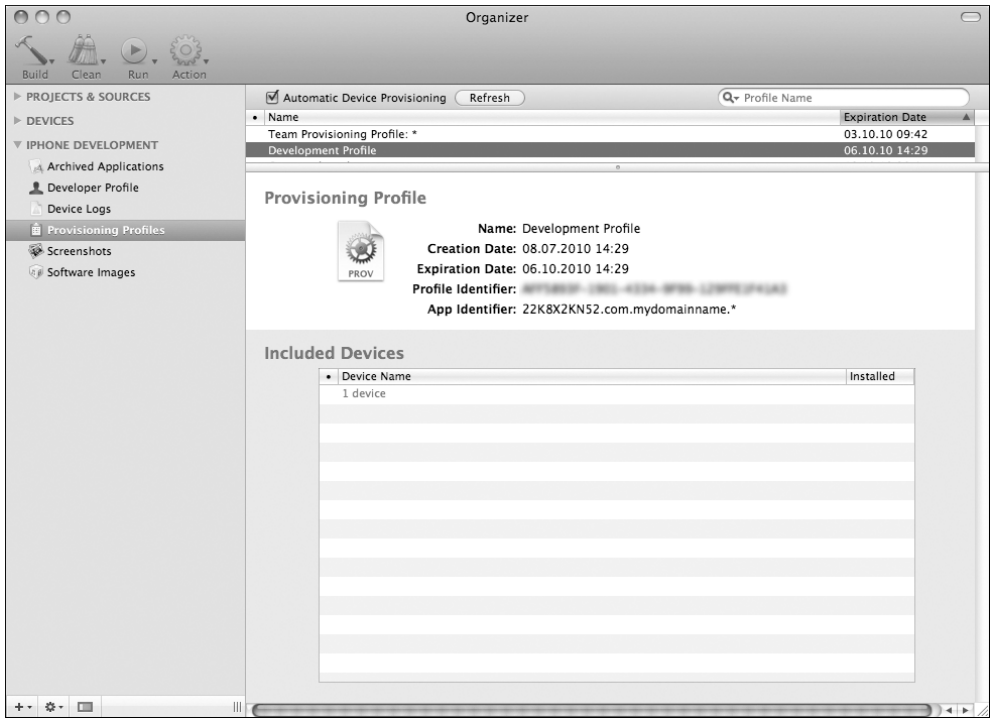
Device Logs visualisiert die unter dem Pfad *<user home>/Library/Logs/Crash Reporter/MobileDevice/* befindlichen Crashlogs. Dort speichert iTunes bei der Synchronisation die auf den Geräten gefundenen Fehlerberichte. Gerade bei mehreren Testgeräten spielt diese Ansicht ihren Vorteil der Übersichtlichkeit aus. Mit einem Filter lassen sich die Berichte einschränken und mit der Maus und der delete-Taste schnell unerwünschte Berichte von Anwendungen von Drittanbietern entfernen.



Schnellen Zugriff auf die Crashlogs aus iTunes verspricht die Ansicht Device Logs.

Provisioning Profiles

Eine Übersicht über alle Profile auf dem Rechner zeigt *Provisioning Profiles* an. Neben reinen Verwaltungsaufgaben wie das Löschen von Profilen zeigt sie den Status jedes Profils und Informationen zu den installierten Testgeräten an. So können Karteileichen effizient herausgefiltert und gelöscht werden – gerade bei vielen Profilen ist das oft ein Segen.



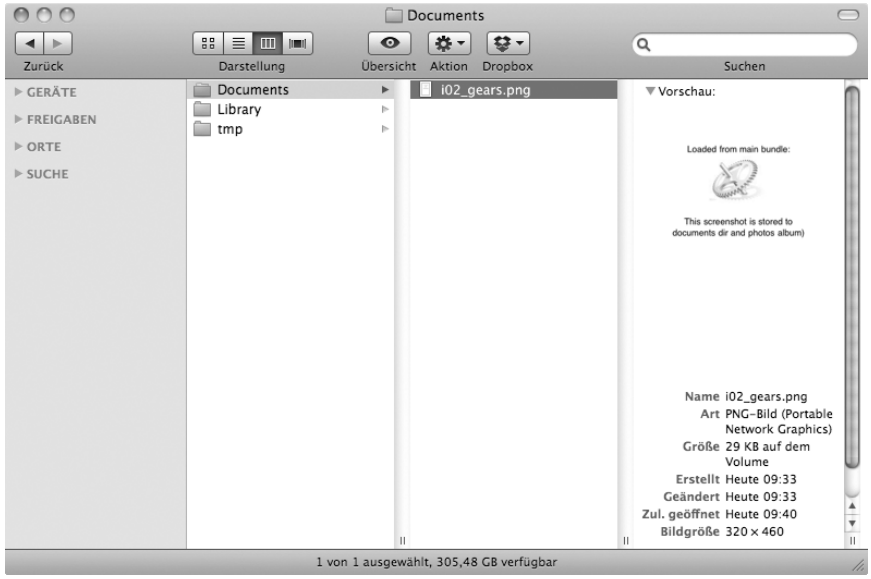
Screenshots

Die aus dem Organizer aufgenommenen Screenshots werden hier geräteübergreifend zusammengefasst. Jedes der Bilder kann mit einem Klick auf den Button *Save as Default image* zum Startbildschirm eines der geöffneten Projekte erkoren werden. Und so bekommen Sie einen Screenshot in den Organizer:

1. Schließen Sie ein Endgerät per USB an den Rechner an und warten Sie, bis der Organizer es unter *DEVICES* als aktiv führt – dargestellt durch einen grünen Punkt.
2. Klicken Sie unter *DEVICES* auf das Gerät und wählen Sie das Register *Screenshots*.
3. Mit dem Button *Capture* nehmen Sie den aktuellen Bildschirm des Gerätes auf. Auch hier haben Sie gleich die Möglichkeit, das Bild als Startbildschirm für ein Projekt zu bestimmen.
4. Die Screenshots aller Geräte finden Sie sofort auch im Bereich *IPHONE DEVELOPMENT*.

Software Images

Die als Software Images bzw. Updates bezeichneten Firmwareversionen lassen sich über den Organizer verwalten. Per Drag & Drop können die Beta- und Vollversionen mit der Dateiendung *ipsw* in das Fenster gezogen werden. Dort werden die verfügbaren Dateien nach Version und Gerätetyp aufgeschlüsselt angezeigt. Im Register *Summary* unter *DEVICES* kann für jedes angeschlossene Gerät aus der Liste eine verfügbare Firmware ausgewählt und mit dem Button *Restore iPhone* installiert werden. Dabei wird das Gerät komplett zurückgesetzt und alle Inhalte werden gelöscht. Ein Backup mit den aktuellen Daten ist also dringend anzuraten.



Die im Dokumentverzeichnis gespeicherte Datei kann über den Organizer auf die Festplatte kopiert und danach im Finder betrachtet werden.

31.3 Workshop: Animationen mit Bildern

Neben den bisher in Kapitel 27 vorgestellten Animationen werden oft auch komplexere Animationen benötigt, wie sie z. B. in animierten GIF-Grafiken vorkommen. Das iPhone-SDK unterstützt dieses animierte Grafikformat nicht. Stattdessen gibt es die Möglichkeit, durch ein Aneinanderreihen von Einzelbildern einen vergleichbaren Effekt zu erzielen. Die Klasse *UIImageView* bietet dafür drei Eigenschaften zur Konfiguration an:

- `@property(nonatomic, copy) NSArray *animationImages;`
- `@property(nonatomic) NSTimeInterval animationDuration;`
- `@property(nonatomic) NSInteger animationRepeatCount;`

Die einzelnen Grafiken der Animation werden als Array *animationImages* dem *UIImageView* übergeben. Die Dauer der Animation wird mittels *animationDuration* eingestellt. Das heißt, die Bildrate errechnet sich somit aus *animationDuration/[animationImages count]*. Wird keine *animationDuration* gesetzt, wird von einer Rate von 30 Frames pro Sekunde ausgegangen. Das ist auch die vernünftige obere Grenze der Framerate, die auf allen Geräten zuverlässig abgespielt wird. *animationRepeatCount* stellt die Anzahl der Wiederholungen der Animation ein. Ein Wert von 0 wiederholt sie endlos.

Die so definierte Animation wird mit der Methode *startAnimating* gestartet und kann durch *stopAnimating* beendet werden. Im folgenden Workshop wird eine Animation mit acht Einzelbildern als Ladeanzeiger gestartet:

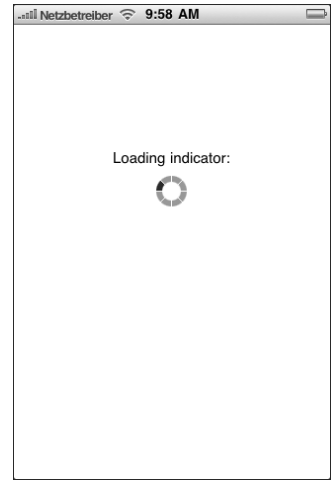
```
@interface WorkshopViewController : UIViewController {
    IBOutlet UIImageView *loadingIndicatorImageView;
}
@property(nonatomic, retain) IBOutlet UIImageView
    *loadingIndicatorImageView;

@end
```

Beispiel 31.3: *WorkshopViewController.h*

```
@implementation WorkshopViewController
@synthesize loadingIndicatorImageView;

- (void)viewDidAppear:(BOOL)animated {
    [super viewDidAppear:animated];

    // now load the images into the outlets

    NSArray *animationImages = [NSArray arrayWithObjects:
        [UIImage imageNamed:@"loading_01.png"],
        [UIImage imageNamed:@"loading_02.png"],
        [UIImage imageNamed:@"loading_03.png"],
        [UIImage imageNamed:@"loading_04.png"],
        [UIImage imageNamed:@"loading_05.png"],
        [UIImage imageNamed:@"loading_06.png"],
        [UIImage imageNamed:@"loading_07.png"],
        [UIImage imageNamed:@"loading_08.png"],
        nil];

    // configure animation
    loadingIndicatorImageView.animationImages = animationImages;
    loadingIndicatorImageView.animationDuration = .8f;
    loadingIndicatorImageView.animationRepeatCount = 0;

    [loadingIndicatorImageView startAnimating];
}

- (void)dealloc {
    [loadingIndicatorImageView release];
    [super dealloc];
}

@end
```

Beispiel 31.3: *WorkshopViewController.m*

Der Workshop mit seinem alternativen Ladeanzeiger stellt sich wie abgebildet dar.



Achtung: Speicherengpässe

Die Animation durch *UIImageView* lädt die Bilder vorab in den Speicher und hält sie während der gesamten Animation vor. Dadurch kann es bei größeren Grafiken oder einer Vielzahl an Einzelbildern zu Speicherengpässen kommen. Eine Lösung dafür wird im nächsten Workshop vorgestellt.

31.4 Workshop: Speicheroptimierte Bildanimation

Die im vorherigen Abschnitt vorgestellte Möglichkeit der Bildanimation wird in den meisten Fällen den Bedürfnissen genügen. Bei ganzformatigen Animationen oder Animationen mit einer großen Anzahl an Einzelbildern ist das Laden der Animation zu Beginn nachteilig, weil dadurch der maximale Speicherbedarf während der kompletten Animation bestehen bleibt.

Optimieren kann man das Verhalten durch eine eigene Subklasse der Klasse *UIImageView*. Anstatt das Array *animationImages* zu nutzen, wird eine eigene Instanzvariable *animationImageFilenames* eingeführt. Diese enthält die Pfade zu den einzelnen Bildern. Während der Animation werden die Bilder nach und nach mit der Methode *imageWithContentsOfFile:* geladen. So ist nur jeweils der aktuell dargestellte Teil der Animation im Speicher. Danach werden die Methoden *startAnimating* und *stopAnimating* überschrieben. Sie steuern zukünftig für jedes Einzelbild die Animation, laden das entsprechende Bild nach und starten einen *NSTimer* für den Aufruf der nächsten Animationssequenz.

Allerdings hat diese Methode auch ein paar Nachteile: Durch das Nachladen während der Animation wird die maximal mögliche Abspielrate von etwa 30 Frames pro Sekunde auf ca. 10 bis 15 reduziert, abhängig von der Leistung des Endgerä-