4 Ihre erste App

In diesem Abschnitt werden Sie Ihre ersten Schritte in der Windows-Phone-8-Entwicklung machen. Dabei werden Sie eine modifizierte »Hallo Welt«-Anwendung schreiben. Daneben lernen Sie das Testen Ihrer Anwendung sowie die Unterstützung von Hoch- und Querformat kennen.

4.1 Die Projektstruktur

Öffnen Sie Visual Studio und legen Sie ein neues Projekt vom Typ Windows-Phone-Anwendung an. Vergeben Sie als Projektnamen *HalloWelt* und erstellen Sie das Projekt, indem Sie auf *OK* klicken. Es werden einige Elemente erzeugt und es öffnet sich der Designer, mit dem Sie Ihre erste Seite entwerfen können.

Im Projektmappen-Explorer werden die erzeugten Dateien angezeigt (vgl. Abb. 3–8 aus Abschnitt 3.2.6).

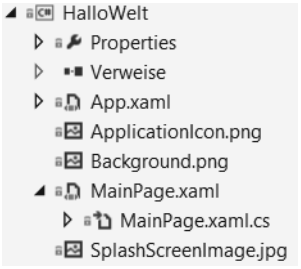

Abb. 4–1 *Projektmappen-Explorer*

Verweise binden weitere DLL-Dateien ein, um dem Projekt zusätzliche Funktionalitäten zur Verfügung zu stellen, diese weiterzuverwenden oder zusätzliche Steuerelemente hinzuzufügen.

Die Datei *App.xaml* mit ihrer zugehörigen Code-Behind-Datei definiert anwendungsweite Ressourcen und initialisiert die Anwendung für den Start. Des

Weiteren wird hier Code hinterlegt, um das Tombstoning (siehe Kapitel 7) zu unterstützen.

Die Grafiken *ApplicationIcon.png* und *Background.png* definieren die Bilder, die in der Programmliste bzw. für die Anwendungskachel (wenn der Benutzer die Anwendung zur Startseite hinzufügt) verwendet werden.

4.2 MainPage.xaml – die Standardseite

Eine Windows-Phone-8-Anwendung besteht aus einer oder mehreren Seiten. Jede einzelne Seite stellt eine Klasse dar, die von *PhoneApplicationPage* erbt. Zwischen den Seiten kann ähnlich einer HTML-Seite navigiert werden.

Beim Starten der Anwendung wird zur Standardseite *MainPage.xaml* navigiert. Visual Studio öffnet diese automatisch, wenn Sie ein Windows-Phone-8-Projekt erstellen. Die Seite ist so weit vorbereitet, dass sie dem Modern-UI-Design folgt und nur noch angepasst werden muss.

4.3 Die Oberfläche einer Seite gestalten

Für ein korrektes Layout der einzelnen Elemente in Hoch- und Querformat verwenden Sie passende Layoutcontainer. Diese und andere Seitenelemente werden Ihnen in Kapitel 5 detaillierter vorgestellt. Für Ihre erste App werden Sie folgende Elemente kennenlernen:

- Grid: ein Layoutcontainer, um Elemente tabellarisch anzuordnen
- TextBlock: Darstellung von nicht änderbarem Text
- TextBox: Eingabefeld für Text
- Button: eine Schaltfläche

Nun beginnt die Definition der einzelnen Zeilen. Dafür wählen Sie den leeren Seitenbereich an, womit Sie das Grid `ContentGrid` ausgewählt haben. Es erscheint eine hellblaue Umrandung, über die Sie Zeilen und Spalten definieren können (siehe Abb. 4–2).

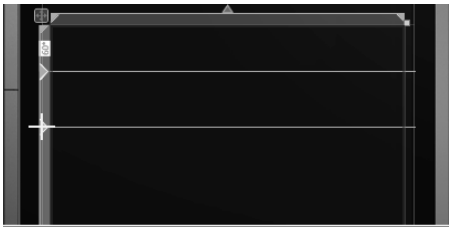

Abb. 4–2 *Grid-Zeilen definieren im Designer*

Klicken Sie zum Erzeugen einer Zeile auf den hellblauen Rand auf der linken Seite des Grids.

Beobachten Sie, wie Visual Studio das notwendige XAML generiert. Dieses können Sie alternativ auch direkt eingeben und der Designer aktualisiert sich entsprechend. Definieren Sie insgesamt drei Zeilen.

Nachdem die Zeilen definiert sind, können Sie nun Elemente hinzufügen. Ziehen Sie einen TextBlock aus dem *Werkzeugkasten*-Fenster in die erste Zeile. Dieser wird dort platziert, wo Sie die Maustaste losgelassen haben, und verfügt über feste Größen und Abstände. Diese erschweren allerdings ein korrektes Layout bei Hoch- und Querformat. Löschen Sie diese Angaben, indem Sie im Kontextmenü des TextBlocks den Punkt *Layout zurücksetzen* → *Alle* wählen (siehe Abb. 4–3). Nun nimmt der TextBlock die komplette Zeilengröße ein.

Hinweis

Wenn Sie Elemente per Drag & Drop auf den Designer ziehen, benennt Visual Studio diese Elemente automatisch. Diese sind im Gegensatz zu Windows Forms nicht zwingend notwendig. Sie benötigen die Namen nur, wenn das Element per Code angesprochen werden soll.

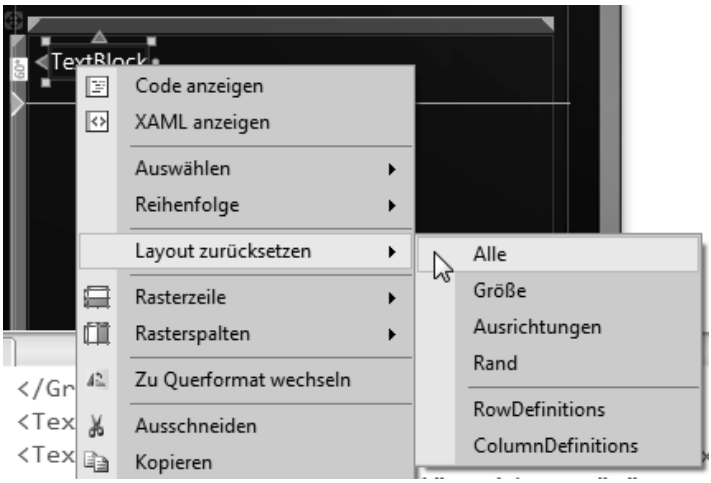

Abb. 4–3 *Layout zurücksetzen*

Ordnen Sie nun darunter je eine TextBox und einen Button an und lassen Sie diese ebenfalls die komplette Zeile füllen.

Nachdem alle Elemente definiert sind, können diese noch über ihre Eigenschaften angepasst werden, um verschiedene Aspekte festzulegen. Darunter fallen u.a.

- Name, unter dem das Element im Programmcode ansprechbar ist,
- Text, der angezeigt werden soll, sowie horizontale und vertikale Ausrichtung.

Zur Navigation zwischen den Elementen können Sie neben dem Designer auch den XAML-Quelltext verwenden oder das Fenster *Dokumentgliederung*. Folgende Elemente sollen bearbeitet werden:

Element	Eigenschaft	Wert
ApplicationTitle	Text	ERSTE ANWENDUNG
PageTitle	Text	Hallo Welt
textBlock1	Text	Name:
button1	Content	Klick mich!

Tab. 4–1 *Elementeigenschaften für »Hallo Welt«*

Als letzten Designschritt passen Sie die Zeilenhöhen so an, dass diese ausreichend hoch sind, um ihren Inhalt darzustellen. Zeigen Sie dafür auf die Zeilenhöhe im Designer und wählen Sie in der erscheinenden Auswahl *Auto* (siehe Abb. 4–4).

Abb. 4–4 *Zeilenhöhe anpassen*

4.4 Programmlogik hinzufügen

Nun fehlt nur noch die Angabe, was beim Klicken des Buttons passieren soll.

Wenn ein Button geklickt wird, wird ein Ereignis ausgelöst, das über die Statusänderung informiert. Im Falle eines Klicks wird das Ereignis *Click* ausgelöst. Um darauf zu reagieren, muss ein Ereignis- bzw. Eventhandler definiert werden. Dies können Sie über das Eigenschaftenfenster realisieren, über XAML oder durch einen einfachen Doppelklick auf den Button.

Visual Studio erzeugt sowohl den Eventhandler (dies ist eine gewöhnliche Methode) als auch das XAML, um die Verknüpfung zwischen Button und Eventhandler herzustellen (siehe Abb. 4–4). In Listing 4–2 sehen Sie die Codezeilen, die Sie ergänzen müssen.

```
01 <Button Click="button1_Click" … />
```

Listing 4–1 *Die generierte Verknüpfung zwischen XAML und Code*

```
01 private void button1_Click(object sender, RoutedEventArgs e)
02 {
03   MessageBox.Show("Hallo" + textBox1.Text);
04 }
```

Listing 4–2 *Der Eventhandler für das Click-Ereignis des Buttons*

4.5 Anwendung ausführen und testen

Damit Ihre Anwendung ausgeführt werden kann, müssen Sie diese für Windows Phone 8 aufbereiten. Dies erledigt Visual Studio automatisch, wenn Sie das Projekt erstellen (über den Menüpunkt *Debuggen* oder durch Drücken von F6). Visual Studio führt folgende Aufgaben dabei aus:

- Überprüfung Ihres Codes auf Syntaxfehler
- Kompilieren Ihres Codes
- Verpacken Ihres Projektes mit allen Ressourcen zu einem Windows-Phone-8-Anwendungspakets. Dies ist eine Datei mit der Endung XAP.

Wenn Sie kein Windows-Phone-8-Gerät besitzen, bietet sich zum Testen der Emulator an. Klicken Sie dazu einfach auf den grünen Pfeil in der Symbolleiste (siehe Abb. 4–5).

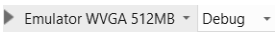

Abb. 4–5 *Testvorgang starten*

Wenn noch nicht geschehen, wird Ihr Projekt jetzt erstellt, der Emulator wird gestartet und die Anwendung wird übertragen sowie gestartet (siehe Abb. 4–6). Testen Sie nun Ihre erste Anwendung.

Abb. 4–6 *Die erste Anwendung wird im Emulator ausgeführt.*

Damit das eigene Gerät zum Testen verwendet werden kann, muss dieses freigeschaltet werden. Der Freischaltprozess ist dank des Tools *Windows Phone Developer Registration* (Sie finden es im Startmenü unter den *Windows Phone SDK 8.0*) sehr einfach. Der Freischaltprozess ist detailiert in Kapitel 56 beschrieben.

4.6 Hoch- und Querformat unterstützen

Als Entwickler müssen Sie explizit festlegen, welche Ausrichtung Sie unterstützen. Standardmäßig ist dies nur das Hochformat. Um sowohl das Hoch- als auch das Querformat zu unterstützen, müssen Sie lediglich die Eigenschaft *SupportedOrientation* der Seite ändern (siehe Listing 4–3). Testen Sie dies mit Ihrer ersten Anwendung.

```
01 <phone:PhoneApplicationPage ...
    SupportedOrientations="PortraitOrLandscape" />
```

Listing 4–3 *Festlegung der unterstützten Seitenausrichtungen*

4.7 WMAppManifest.xml – die erste Konfiguration

Soll Ihre Startseite anders heißen als MainPage.xaml oder bevorzugen Sie eine andere Projektstruktur, müssen Sie festlegen, wo sich Ihre Startseite befindet. Dazu bearbeiten Sie die Datei *WMAppManifest.xml*, die Sie im *Properties*-Ordner im Projektmappen-Explorer finden.

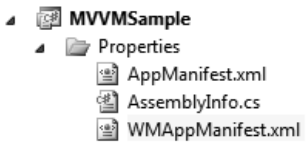

Abb. 4–7 *Die WMAppManifest.xml*

In der *WMAppManifest.xml* kann das Attribut *NavigationPage* innerhalb des Elements *DefaultTask* auf eine beliebige XAML-Seite gelegt werden.

```
01 <Tasks>
02   <DefaultTask
03     Name = "default"
04     NavigationPage="Views/MyOwnStartView.xaml"/>
05 </Tasks>
```

Listing 4–4 *Konfiguration der WMAppManifest.xml*

Neben der Startseite kann auch der Pfad des App-Icons und des Startbildschirms geändert werden.