Chapter 10
Tiles and Printing

With such a rich and varied API as the one provided by TouchDevelop, there are naturally some features which are very hard to group with other related material. This chapter covers two topics which do not seem to belong in the other chapters. The first topic is ‘Tiles’, an important visual feature of the Windows phone user interface. The second topic is access to a printer via a home server.

10.1 Tiles 141
10.2 Printing via a Home Server 147

10.1 Tiles

The home screen of the phone is populated with tiles. Tapping a tile will invoke a program associated with that tile. The user can customize the phone’s start screen by selecting which programs have tiles and by re-arranging the layout of the tiles.

An important concept is that of pinning a tile. Pinning the tile simply means creating a tile on the home screen. Unpinning the tile means removing the tile. Holding one’s finger on a tile for a second or two causes the tile to expand in size and an unpin icon appear in the top right corner. Tapping that icon deletes the tile.

Tiles show pictures or icons; they have color; they can display textual information; they can change what is displayed. Most tiles are the standard size, which is a square, whereas some tiles are double size. All this is under the control of the associated program, the program which is invoked when
the tile is tapped.

Tiles always have a front side; they optionally can have a back side. When there is both a front side and a back side, the phone’s software randomly flips the tile over to show the other side.

10.1.1 Tiles in TouchDevelop

Most of the tile’s capabilities can be accessed by a TouchDevelop script. However, it is not possible to create a double sized tile and it is not possible to leave a background TouchDevelop process running to change the images and text displayed on a tile. When the tile is visible, TouchDevelop is not running, and therefore the tile cannot change – other than being randomly flipped over to alternate between the front and back sides.

A tile can be associated with an action in a TouchDevelop script. If the script contains more than one publicly visible action, there can be more than one tile for the script. Tapping each tile invokes the associated action. If the action requires parameters, the user is prompted to enter values for the action when it starts.

10.1.2 Creating a tile

Before a script can be programmed to display particular content on a tile, the tile needs to be created. It cannot be created by executing actions in the script itself and it cannot be created with the TouchDevelop editor. Producing a script which works with a tile is necessarily a multistage process.

If the action is which is to be associated with the tile is the main action, steps 1 through 3 below can be short-circuited because a row of icons which includes a pushpin icon should already exist at the bottom of the screen. Otherwise, the three steps need to be followed so that a pushpin icon appears.

1. Create the action which is to be invoked when the tile is tapped. The action need not contain any code at this point. The action must not have the ‘private’ box checked in its properties.
2. Run the action to its conclusion. (If necessary halt its execution by tapping the phone's back button.)
3. When the action has finished, a row of icons appears across the
bottom of the phone’s screen. A screenshot of the row of icons appears in Figure 10-1. One of the icons is a pushpin. This icon should be tapped.

4. TouchDevelop is suspended by the above action and the phone’s home screen is displayed. It should now contain a new tile. The front side has the TouchDevelop icon in its top right corner and the name of the action in the bottom left corner; the back side lacks the title but is otherwise identical. Tap the phone’s back button to return to TouchDevelop and tap the back button to return to a list of components for the script being developed. There should now be a new global data item. It has the name $XXX$ tile, where $XXX$ represents the name of the action and its datatype is Tile. It has a value which is an instance of the Tile type.

If all that was wanted was a tile to invoke the script at this action, and if the action already contains the desired code, and if the new tile looks suitable for use, that’s all. However if the script has to display special content on the tile, then the following subsection provides further details.

![Figure 10-1: The ‘Pushpin’ icon](image)

### 10.1.3 Programming the tile’s content

The previous subsection described how to create an instance of the Tile datatype as a global data variable. No other mechanism is provided for creating an instance of Tile.

Once the script contains a variable of type Tile, statements which access the
methods of the Tile type can be added to the script.

**General properties**

A tile should be a square whose sides are 173 pixels in size. The width and height methods should return 173 as their value.

A tile has a background color which will be displayed if no icon image to fill the tile has been provided. If a picture which is smaller in one of its dimensions than 173 pixels is provided as the image, that image is centered in the tile's area and the background color is visible around the edges. If the image is not fully opaque, the background color will show through. The background color is accessed by the background and set background methods.

A user can unpin a tile on the home screen simply by touching the tile for a second or two until a delete option becomes available. The script can repin the tile by employing the pin to start method.

Table 10-1 lists the methods which are not specific to accessing front side or back side content. (As usual, is invalid and post to wall are available but are omitted from the table.)

**Table 10-1: General methods of Tile datatype**

<table>
<thead>
<tr>
<th>Tile Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>background : Color</td>
<td>Returns the tile's background color</td>
</tr>
<tr>
<td>height : Number</td>
<td>Gets the tile height in pixels</td>
</tr>
<tr>
<td>pin to start : Nothing</td>
<td>Pins the tile to the home screen after asking user permission</td>
</tr>
<tr>
<td>width : Number</td>
<td>Gets the tile width in pixels</td>
</tr>
</tbody>
</table>

**Front side properties**

The front side always has a title which is accessed by the title and set title methods. The front side can display an image, which is accessed by the icon, clear icon and set icon methods.

Finally, the front side of the tile can display an integer in the top right corner. That integer must be greater than zero and it can be accessed by the methods counter and set counter. The tile's counter is used by the phone's
mail program and SMS message program to show the number of unread messages. A TouchDevelop script can use it for any purpose whatsoever.

These methods are summarized in Table 10-2.

<table>
<thead>
<tr>
<th>Tile Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear icon : Nothing</td>
<td>Clears the front side icon, if any</td>
</tr>
<tr>
<td>counter : Number</td>
<td>Gets the front side counter</td>
</tr>
<tr>
<td>icon : Picture</td>
<td>Gets the front side icon picture</td>
</tr>
<tr>
<td>set counter(counter : Number) : Nothing</td>
<td>Sets the counter displayed on front side (values ≤ 0 are not displayed)</td>
</tr>
<tr>
<td>set icon(pic : Picture) : Nothing</td>
<td>Sets the front side icon, cropping the picture to the tile size</td>
</tr>
<tr>
<td>set title(title : String) : Nothing</td>
<td>Sets the front side title; if an empty string, the script name is displayed</td>
</tr>
<tr>
<td>title : String</td>
<td>Gets the front side title</td>
</tr>
</tbody>
</table>

**Back side properties**

The back side of the tile shares the same background color as the front side. An icon can be displayed in the center of the tile on top of that background. The icon is accessed by the methods back icon, clear back icon and set back icon.

Textual content can be displayed on the back side. It is accessed via the content and set content methods. The back side of the tile has an optional title, which is accessed by the back title and set back title methods.

These methods are summarized in Table 10-3.

**10.1.4 Demonstration of the tile features**

The script shown in Figure 10-2 uses every one of a tile’s features. When the script is run for the first time, the tile which was created for this script is modified. Figure 10-3 shows the front side and back side of the tile after the script has been run twice causing variable ctr to have the value 2.
Table 10-3: Back side methods of Tile datatype

<table>
<thead>
<tr>
<th>Tile Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>back icon : Picture</td>
<td>Returns the back side icon picture</td>
</tr>
<tr>
<td>back title : String</td>
<td>Returns the back side title</td>
</tr>
<tr>
<td>clear back icon : Nothing</td>
<td>Clears the back side icon, if any</td>
</tr>
<tr>
<td>content : String</td>
<td>Gets the back side content</td>
</tr>
<tr>
<td>set back icon(pic : Picture) : Nothing</td>
<td>Sets the back side icon, cropping the picture to the tile size</td>
</tr>
<tr>
<td>set back title(title : String) : Nothing</td>
<td>Sets the back side title</td>
</tr>
<tr>
<td>set content(content : String) : Nothing</td>
<td>Sets the back side content</td>
</tr>
</tbody>
</table>

Figure 10-2: The ‘Tile Fun’ script

```
action main()
    "Testing all features of a tile"  →  post to wall
    // The front icon is a blue square inside a yellow square
    var front icon := media → create picture(64, 64)
    front icon → clear(colors → yellow)
    front icon → draw rect(10, 10, 44, 44, 0, colors → blue, 3)
    main tile → set background(colors → blue)
    main tile → set content("Back side content")
    main tile → set title("Front title")
    ctr := ctr + 1
    main tile → set counter(ctr)
    // The back icon is a green square holding a brown circle
    var back icon := media → create picture(64, 64)
    back icon → clear(colors → green)
    back icon → fill ellipse(10, 10, 44, 44, 0, colors → brown)
    main tile → set back icon(back icon)
    main tile → set back title("Back title")
```
10.2 Printing via a home server

A home network may include one or more printers. These printers would normally be attached to the network via their WiFi capabilities.

The TouchDevelop script provides two API methods for connecting to a printer. The simplest is the following, which allows the user to select one from a list.

```plaintext
var prtr := home → choose printer
```

An alternative is to use code like the following.

```plaintext
var prtr list := home → printers
if prtr list → count = 0 then
    "No printers found" → post to wall
    time → stop
else
    // use first printer in the list
    var prtr := prtr list → at(0)
```

Once a printer has been selected, various methods are provided for interrogating and using the printer. These methods are listed in Table 10-4.
Table 10-4: Methods of Printer datatype

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>device : Device</td>
<td>Gets detailed information about this device</td>
</tr>
<tr>
<td>is idle : Boolean</td>
<td>Returns true if a new print job would start immediately</td>
</tr>
<tr>
<td>is processing : Boolean</td>
<td>Returns true if the printer is busy</td>
</tr>
<tr>
<td>is stopped : Boolean</td>
<td>Returns true if the printer is stopped and no print jobs can be processed (it needs manual intervention)</td>
</tr>
<tr>
<td>name : String</td>
<td>Gets the name of the printer</td>
</tr>
<tr>
<td>print text(text : String) : Nothing</td>
<td>Queues a new print job to the printer</td>
</tr>
<tr>
<td>state reason : String</td>
<td>Gets a reason for the printer being in its current state</td>
</tr>
</tbody>
</table>