Mit der Programmierumgebung (IDE) Visual Studio 2017 wird eine Anwendung (App) für den Raspberry geschrieben, compiliert und auf das Board übertragen. Dabei ist die IDE bereits für den Raspberry eingerichtet und das Board wird über ein Ethernet vom Laptop erkannt.

In diesem Arbeitsblatt wird zusätzlich noch die Benutzung von Visual Studio beschrieben. Bei den weiteren Projekten wird mehr die Programmierung und die Hardwareumgebung des Raspberry betrachtet.

Wir starten Visual Studio mit *Datei → Neu → Projekt*. Im linken Bereich des Fensters wählen wir *Visual Basic → Windows → Windows IoT Core*. Findet man diesen Eintrag nicht, müssen die *IoT Core Projekt Templates für Visual Studio 2017 eingelesen werden (siehe Arbeitsblatt).* Im mittleren Bereich kennzeichnen wir die Vorlage *Background Application (IoT)*.

Dann wählen wir *Visual Basic → Windows → Universell* und kennzeichnen hier *Leere App*. Wir geben unserer App noch einen sinnvollen Namen und warten, bis Windows das Projekt eingerichtet hat.

Die Projektmappe unserer App steht rechts im Fenster. **Wir rechtsklicken auf** *Verweise → Verweise hinzufügen → Universal Windows → Erweiterungen* und versehen *Windows IoT Extensions für the UWP* mit einem Haken → *ok*.

Wir doppelklicken im Projektmappen-Explorer auf den Eintrag *MainPage.xanl*. Der Designer wird gestartet. Es erscheinen ein Rahmen, der die Seite (page), die später angezeigt wird, darstellt und eine weißes Rechteck, die Benutzeroberfläche (GUI für *Graphical User Interface*).

Jetzt bauen wir aus den Steuerelementen unsere Anwendung zusammen. Wir holen sie aus der Tool-Box auf der linken Seite und lassen sie auf der Benutzeroberfläche fallen. Der Zusammenbau wird in Tabellenform unten kurz beschrieben.

Wie bringen wir unsere App zum Laufen? In der oberen Menueleiste stellen wir die Vorgaben für das Kompilieren und Ausführen der App ein. Dort steht *Debug* oder *Release*, die verwendete Systemarchitektur *x86* oder *x64* und das System, auf dem die App gestartet wird. In unserem Fall ist das *Lokaler Computer*, also der Laptop mit dem wir gerade arbeiten. Klicken wir auf *Lokaler Computer*, wird kompiliert und gestartet. Danach finden wir im Startfenster unseres Laptops unter *Zuletzt hinzugefügt* unsere App.

Wie bringen wir unsere App auf den Raspberry? Wir ändern in der oberen Menueleiste die Systemarchitektur zu *ARM*, der Architektur des Raspberry. Das System, auf dem die App gestartet wird, hat sich zu *Remotecomputer* geändert. Wird der Raspberry vom PC erkannt, müssen wir das Zielsystem noch betätigen. Wir können den Remotecomputer auch per Hand auswählen *Menueleiste → Projekt → Eigenschaften*.

Beim Klicken der Taste *Remotecomputer* wird unsere App übersetzt, auf den Raspberry übertragen und dort ausgeführt. Es erscheint dann über der graphischen Benutzeroberfläche (GUI) von Visual Studio die Taste *Ereignisse des Lebenzyklus*. Hier können wir den Raspberry vom PC aus steuern z.B. Anhalten.

Welche Apps sind auf dem Raspberry installiert? Die Apps lassen sich vom PC aus unter ihrem Paketnamen anzeigen, z.B. mit der Web-Bedienoberfläche (http:// 192.168.0.103:8080). Mit einem Klick auf *Start* können wir die markierte App ausführen. Klicken wir auf die Schaltfläche *Set Default* wird die markierte App bei jedem Neustart sofort auf dem Raspberry gestartet.

Ort	Eintrag	Vorgang	Auswirkung
Menüleiste oben	*Datei -> Neu -> Projekt*		Ein neues Projekt wird gebildet.
linker Bereich	*Visual Basic -> Windows -> Windows IoT Core -> Background Application (IoT)*	markieren	Die Programmiersprache wird ausgewählt.
linker Bereich	*Visual Basic -> Windows -> Universell -> Leere App*	markieren	
unten	Erster Versuch	eingeben	Die App erhält einen sinnvollen Namen.
Projektmappenexplorer	*Verweise* [rechtsklicken] *Verweise hinzufügen*	anklicken	Die Steuerelemente werden angepasst.
links	*Universal Windows -> Erweiterungen -> Windows IoT Extensions for the UWP (Universelle Windows Plattform)*	ok	
Projektmappenexplorer	*MainPage.xaml*	klicken	Der Designer öffnet die MainPage, die graphische Programmieroberfläche (GUI) in der Mitte
MainPage	Benutzeroberfläche, *Grid* (Weißer Bereich)	anklicken	
Eigenschaftsfenster rechts unten			Für dieses Objekt vom Typ Grid können die Eigenschaften (z.B. Layout) eingestellt werden
MainPage	Rahmen um die Benutzeroberfläche, *Page*	anklicken	
Eigenschaftsfenster rechts unten			Für dieses Objekt vom Typ Page können die Eigenschaften (z.B. Layout) eingestellt werden
Toolbox, linker Fensterrand	*Toolbox -> Alle XAML-Steuerelemente*	anklicken	
Toolbox	*Textblock*	in die Benutzeroberfläche ziehen	
Eigenschaftsfenster	*Überschrift*	eingeben	Der Textblock erhält den Namen *Überschrift*
Eigenschaftsfenster	*Allgemein -> Text*		Die sichtbare Zeichenkette *Erste GMG-App* wird eingeben.
Ein weiterer Textblock mit Namen *Anzeige* und Inhalt *Hallo Welt* wird auf die Benutzeroberfläche gebracht.			

Ort	Eintrag	Vorgang	Auswirkung
Toolbox	*Button*	in die Benutzer-oberfläche ziehen	Der Schaltknopf *Drück mich* auf der Benutzeroberfläche platziert.
Eigenschaftsfens-ter	*Allgemein -> Content*		die sichtbare Zeichenkette D*rück mich* eingeben.
Projektmappen-explorer	*MainPage.xaml.vb*	doppeltkli-cken	Den Code-Editor öffnen.
Reiter der Sour-cecode-Fenster	*MainPage Ereignisse*	anklicken	Ein neuer Ereignishändler wird im Source-Code gene-riert.
Reiter des Sour-cecode-Fenster	*Loading*	anklicken	Er tritt beim Laden der App in Aktion.
Souce Code	*Anzeige.Visibility = Visibility.Collapsed*	einfügen	Der Text wird beim Pro-grammstart nicht gezeigt.
Benutzeroberflä-che		Auf den Button doppelkli-cken	Der Sourcocode wird ange-zeigt.
Source Code	*Anzeige.Visibility = Visibility.Visible*	einfügen	Der Code wird beim Anklicken des Buttons ausgeführt.
obere Menüleiste	*Lokaler Computer*	anklicken	Der Code der Benutzerober-fläche wird kompiliert und auf dem Laptop ausgeführt.
	Zwischendurch muss das Projekt immer wieder gespeichert werden.		