

Awesome Dictionary

Bir veya birden çok sözlüğü dahil edip kelimelerin anlamlarini bulmaya yarayan bir uygulama

Table of Contents

Symbol Reference	1
AwesomeDictionary Namespace	1
AwesomeDictionary.Resources Namespace	2
Classes	2
AppResources Class	2
Classes	16
AboutPage Class	17
AboutPage>AboutPage Constructor	17
All Class	18
All Properties	18
AlmancaTurkce Class	19
AlmancaTurkce Properties	20
AlphaKeyGroup Class	21
AlphaKeyGroup.AlphaKeyGroup Constructor	21
AlphaKeyGroup Methods	21
AlphaKeyGroup Properties	22
AlphaKeyGroup Delegates	23
App Class	23
App.App Constructor	23
App Methods	24
App Properties	25
AppSettings Class	25
AppSettings Properties	25
AwesomeDictionaryDataContext Class	26
AwesomeDictionaryDataContext.AwesomeDictionaryDataContext Constructor	27
AwesomeDictionaryDataContext Fields	27
BackgroundColorSettingsPage Class	29
BackgroundColorSettingsPage.BackgroundColorSettingsPage Constructor	30
BackgroundColorSettingsPage Methods	30
BilisimSozlugu Class	31
BilisimSozlugu Properties	32
BuyukLugat Class	33
BuyukLugat Properties	33
EnglishTurkishVol1 Class	34
EnglishTurkishVol1 Properties	34
EnglishTurkishVol2 Class	35

EnglishTurkishVol2 Properties	35
Favourite Class	36
Favourite Properties	37
FontFamilySettingsPage Class	37
FontFamilySettingsPage.FontFamilySettingsPage Constructor	38
FontFamilySettingsPage Methods	38
FontSizeSettingsPage Class	39
FontSizeSettingsPage.FontSizeSettingsPage Constructor	40
FontSizeSettingsPage Methods	40
GeneralSettingsPage Class	41
GeneralSettingsPage.GeneralSettingsPage Constructor	42
GeneralSettingsPage Methods	45
KelimeAnlamlari Class	46
KelimeAnlamlari Properties	46
LanguageSettingsPage Class	47
LanguageSettingsPage.LanguageSettingsPage Constructor	47
LanguageSettingsPage Methods	48
LocalizedStrings Class	49
LocalizedStrings Properties	49
MainPage Class	49
MainPage.MainPage Constructor	49
NameDetailPage Class	50
NameDetailPage.NameDetailPage Constructor	51
NameDetailPage Fields	52
NameDetailPage Methods	53
OxfordEnglishEnglish Class	54
OxfordEnglishEnglish Properties	55
RisaleNur Class	56
RisaleNur Properties	56
SearchPage Class	57
SearchPage.SearchPage Constructor	57
SearchPage Methods	58
StatisticsPage Class	58
StatisticsPage.StatisticsPage Constructor	59
StatisticsPage Methods	59
Files	59
AboutPage.xaml.cs	60
All.cs	63
AlmancaTurkce.cs	64
AlphaKeyGroup.cs	64

App.xaml.cs	66
AppResources.Designer.cs	73
AppSettings.cs	87
AssemblyInfo.cs	87
AwesomeDictionary.csproj	88
AwesomeDictionary.sln	88
AwesomeDictionaryDataContext.cs	88
BackgroundColorSettingsPage.xaml.cs	89
BilisimSozlugu.cs	92
BuyukLugat.cs	93
EnglishTurkishVol1.cs	94
EnglishTurkishVol2.cs	94
Favourite.cs	95
FontFamilySettingsPage.xaml.cs	96
FontSizeSettingsPage.xaml.cs	98
GeneralSettingsPage.xaml.cs	101
KelimeAnlamlari.cs	120
LanguageSettingsPage.xaml.cs	121
LocalizedStrings.cs	125
MainPage.xaml.cs	125
NameDetailPage.xaml.cs	129
OxfordEnglishEnglish.cs	139
RisaleNur.cs	139
SearchPage.xaml.cs	140
StatisticsPage.xaml.cs	144

Index

a

1 Symbol Reference

























1.1 AwesomeDictionary Namespace

This is namespace AwesomeDictionary.

Namespaces

Name	Description
Resources (see page 2)	This is namespace AwesomeDictionary.Resources.

Classes

	Name	Description
	AboutPage (see page 17)	This is class AwesomeDictionary>AboutPage.
	All (see page 18)	This is class AwesomeDictionary.All.
	AlmancaTurkce (see page 19)	This is class AwesomeDictionary.AlmancaTurkce.
	AlphaKeyGroup (see page 21)	This is class AwesomeDictionary.AlphaKeyGroup.
	App (see page 23)	This is class AwesomeDictionary.App.
	AppSettings (see page 25)	This is class AwesomeDictionary.AppSettings.
	AwesomeDictionaryDataContext (see page 26)	This is class AwesomeDictionary.AwesomeDictionaryDataContext.
	BackgroundColorSettingsPage (see page 29)	This is class AwesomeDictionary.BackgroundColorSettingsPage.
	BilisimSozlugu (see page 31)	This is class AwesomeDictionary.BilisimSozlugu.
	BuyukLugat (see page 33)	This is class AwesomeDictionary.BuyukLugat.
	EnglishTurkishVol1 (see page 34)	This is class AwesomeDictionary.EnglishTurkishVol1.
	EnglishTurkishVol2 (see page 35)	This is class AwesomeDictionary.EnglishTurkishVol2.
	Favourite (see page 36)	This is class AwesomeDictionary.Favourite.
	FontFamilySettingsPage (see page 37)	This is class AwesomeDictionary.FontFamilySettingsPage.
	FontSizeSettingsPage (see page 39)	This is class AwesomeDictionary.FontSizeSettingsPage.
	GeneralSettingsPage (see page 41)	This is class AwesomeDictionary.GeneralSettingsPage.
	KelimeAnlamlari (see page 46)	This is class AwesomeDictionary.KelimeAnlamlari.
	LanguageSettingsPage (see page 47)	This is class AwesomeDictionary.LanguageSettingsPage.
	LocalizedStrings (see page 49)	Provides access to string resources.
	MainPage (see page 49)	This is class AwesomeDictionary.MainPage.
	NameDetailPage (see page 50)	This is class AwesomeDictionary.NameDetailPage.
	OxfordEnglishEnglish (see page 54)	This is class AwesomeDictionary.OxfordEnglishEnglish.
	RisaleNur (see page 56)	This is class AwesomeDictionary.RisaleNur.
	SearchPage (see page 57)	This is class AwesomeDictionary.SearchPage.
	StatisticsPage (see page 58)	This is class AwesomeDictionary.StatisticsPage.

1.1.1 AwesomeDictionary.Resources Namespace

This is namespace AwesomeDictionary.Resources.

Classes

	Name	Description
	AppResources (🔗 see page 2)	A strongly-typed resource class, for looking up localized strings, etc.

1.1.1.1 Classes

The following table lists classes in this documentation.

Classes

	Name	Description
	AppResources (🔗 see page 2)	A strongly-typed resource class, for looking up localized strings, etc.

1.1.1.1.1 AppResources Class

A strongly-typed resource class, for looking up localized strings, etc.

Class Hierarchy

 AwesomeDictionary.Resources.AppResources
--

C#

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.StronglyTypedResourceBuilder",
"4.0.0.0")]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
public class AppResources;
```

File

AppResources.Designer.cs (🔗 see page 73)




Description

This class was auto-generated by the StronglyTypedResourceBuilder class via a tool like ResGen or Visual Studio. To add or remove a member, edit your .ResX file then rerun ResGen with the /str option, or rebuild your VS project.















Methods
























	Name	Description
	AppResources (🔗 see page 5)	This is AppResources, a member of class AppResources.

AppResources Properties

	Name	Description
	About (🔗 see page 6)	Looks up a localized string similar to About.
	AboutAwesomeDictionary (🔗 see page 6)	Looks up a localized string similar to About (🔗 see page 6) Awesome Dictionary.
	AboutTheApp (🔗 see page 6)	Looks up a localized string similar to About (🔗 see page 6) The App (🔗 see page 23).

	AboutTheAppText (see page 6)	Looks up a localized string similar to Hi everybody. I am with you with a new app. I investigate a lot of dictionary applications in Windows Phone and i tried to make a simple app which you will like it. If you rate the app and send your thoughts to coderserdar@outlook.com, I will be so appreciated to you. With my best regards. CoderSerdar.
	AddToFavourites (see page 6)	Looks up a localized string similar to Add To Favourites.
	AtLeastOneDictionary (see page 6)	Looks up a localized string similar to You Should Select (see page 13) At Least One Dictionary.
	Background (see page 6)	Looks up a localized string similar to Background.
	BackgroundColor (see page 6)	Looks up a localized string similar to Background (see page 6) Color.
	BackgroundColorChangedSuccessfully (see page 6)	Looks up a localized string similar to Background (see page 6) Color Has Been Changed Successfully.
	BackgroundImage (see page 7)	Looks up a localized string similar to Background (see page 6) Image.
	BackgroundImageChangedSuccessfully (see page 7)	Looks up a localized string similar to Background (see page 6) Image Has Been Changed Successfully.
	BackgroundImageRemovedSuccessfully (see page 7)	Looks up a localized string similar to Background (see page 6) Image Has Been Removed Successfully.
	BackgroundSettingsResetSuccessfully (see page 7)	Looks up a localized string similar to Background (see page 6) Settings (see page 14) Has Been Reset Successfully.
	Black (see page 7)	Looks up a localized string similar to Black.
	Blue (see page 7)	Looks up a localized string similar to Blue.
	Brown (see page 7)	Looks up a localized string similar to Brown.
	BuyukLugat (see page 7)	Looks up a localized string similar to Buyuk Lugat (Turkish (see page 15)->Turkish (see page 15)).
	Cancel (see page 7)	Looks up a localized string similar to Cancel.
	ComputerDictionary (see page 8)	Looks up a localized string similar to Computer Dictionary (English (see page 8)->Turkish (see page 15)).
	ContactWithUs (see page 8)	Looks up a localized string similar to Contact With Us.
	Culture (see page 8)	Overrides the current thread's CurrentUICulture property for all resource lookups using this strongly typed resource class.
	DictionariesInstalledSuccessfully (see page 8)	Looks up a localized string similar to Dictionaries Have Been Installed Successfully.
	DictionariesUninstalledSuccessfully (see page 8)	Looks up a localized string similar to Dictionaries Have Been Uninstalled Successfully.
	DictionaryInstall (see page 8)	Looks up a localized string similar to Dictionary Install (see page 10).
	English (see page 8)	Looks up a localized string similar to English.
	EnglishTurkishVol1 (see page 8)	Looks up a localized string similar to English (see page 8)->Turkish (see page 15) Dictionary Vol. 1.
	EnglishTurkishVol2 (see page 9)	Looks up a localized string similar to English (see page 8)->Turkish (see page 15) Dictionary Vol. 2.
	ExitApp (see page 9)	Looks up a localized string similar to Exit App (see page 23).
	ExitAppQuestion (see page 9)	Looks up a localized string similar to Are You Sure To Exit The Application?.
	FavouriteNameCount (see page 9)	Looks up a localized string similar to Favourite (see page 36) Word (see page 15) Count.
	Font (see page 9)	Looks up a localized string similar to Font.
	FontFamily (see page 9)	Looks up a localized string similar to Font (see page 9) Family.
	FontFamilyChangedSuccessfully (see page 9)	Looks up a localized string similar to Font (see page 9) Family Has Been Changed Successfully.

	FontSize (see page 9)	Looks up a localized string similar to Font (see page 9) Size.
	FontSizeChangedSuccessfully (see page 9)	Looks up a localized string similar to Font (see page 9) Size Has Been Changed Successfully.
	GeneralSettings (see page 10)	Looks up a localized string similar to General Settings (see page 14).
	German (see page 10)	Looks up a localized string similar to German.
	GermanTurkish (see page 10)	Looks up a localized string similar to German (see page 10)->Turkish (see page 15) Dictionary.
	Gray (see page 10)	Looks up a localized string similar to Gray.
	Green (see page 10)	Looks up a localized string similar to Green.
	Install (see page 10)	Looks up a localized string similar to Install.
	Installing (see page 10)	Looks up a localized string similar to Installing.
	Language (see page 10)	Looks up a localized string similar to Language.
	LanguageWarning (see page 10)	Looks up a localized string similar to You may restart the application for changes will be effect..
	Meaning (see page 11)	Looks up a localized string similar to Meaning.
	MyFavourites (see page 11)	Looks up a localized string similar to My Favourites.
	Ok (see page 11)	Looks up a localized string similar to Ok.
	Orange (see page 11)	Looks up a localized string similar to Orange.
	OxfordDictionary (see page 11)	Looks up a localized string similar to Oxford (English (see page 8)->English (see page 8)).
	Purple (see page 11)	Looks up a localized string similar to Purple.
	RandomWords (see page 11)	Looks up a localized string similar to Random 10 Words.
	Rate (see page 11)	Looks up a localized string similar to Rate.
	Red (see page 11)	Looks up a localized string similar to Red.
	RemoveBackgroundImage (see page 12)	Looks up a localized string similar to Remove Background (see page 6) Image.
	RemoveFromFavourite (see page 12)	Looks up a localized string similar to Remove From Favourites.
	RemoveFromFavouriteQuestion (see page 12)	Looks up a localized string similar to You Will Remove The Word (see page 15) From Favourites. Are You Sure?.
	RemoveFromFavourites (see page 12)	Looks up a localized string similar to Remove From Favourites.
	ResetSettings (see page 12)	Looks up a localized string similar to Reset Settings (see page 14).
	ResourceFlowDirection (see page 12)	Looks up a localized string similar to LeftToRight.
	ResourceLanguage (see page 12)	Looks up a localized string similar to en-US.
	ResourceManager (see page 12)	Returns the cached ResourceManager instance used by this class.
	RisaleNur (see page 12)	Looks up a localized string similar to Risale Nur (Turkish (see page 15)->Turkish (see page 15)).
	Search (see page 13)	Looks up a localized string similar to Search.
	SearchCompleted (see page 13)	Looks up a localized string similar to Search (see page 13) Completed.
	SearchInMeanings (see page 13)	Looks up a localized string similar to Search (see page 13) In Meanings.
	SearchResults (see page 13)	Looks up a localized string similar to Search (see page 13) Results.
	SearchTrimFault (see page 13)	Looks up a localized string similar to Search (see page 13) Criteria Can Not Be Empty.
	Select (see page 13)	Looks up a localized string similar to Select.
	SelectBackgroundColor (see page 13)	Looks up a localized string similar to Select (see page 13) Background (see page 6) Color.

	Selected (see page 13)	Looks up a localized string similar to Selected.
	SelectFontFamily (see page 13)	Looks up a localized string similar to Select (see page 13) Font (see page 9) Family.
	SelectFontSize (see page 14)	Looks up a localized string similar to Select (see page 13) Font (see page 9) Size.
	SelectLanguage (see page 14)	Looks up a localized string similar to Select (see page 13) Language (see page 10).
	SendWithAwesomeDictionaryApp (see page 14)	Looks up a localized string similar to Send With Awesome Dictionary App (see page 23).
	SendWithEmail (see page 14)	Looks up a localized string similar to Send With E-Mail.
	SendWithSMS (see page 14)	Looks up a localized string similar to Send With SMS.
	Settings (see page 14)	Looks up a localized string similar to Settings.
	Share (see page 14)	Looks up a localized string similar to Share.
	Source (see page 14)	Looks up a localized string similar to Source.
	Statistics (see page 14)	Looks up a localized string similar to Statistics.
	Synchronizing (see page 15)	Looks up a localized string similar to Synchronizing.
	SystemFault (see page 15)	Looks up a localized string similar to System Has A Fault. Please Try Again Later..
	TotalNameCount (see page 15)	Looks up a localized string similar to Total Word (see page 15) Count.
	Turkish (see page 15)	Looks up a localized string similar to Turkish.
	Uninstall (see page 15)	Looks up a localized string similar to Uninstall.
	Word (see page 15)	Looks up a localized string similar to Word.
	WordAddedFavouriteSuccessfully (see page 15)	Looks up a localized string similar to Word (see page 15) Has Been Added To Favourites Successfully.
	WordAlreadyFavourite (see page 15)	Looks up a localized string similar to Word (see page 15) Is Already Favourite (see page 36).
	WordAndMeaning (see page 15)	Looks up a localized string similar to Word (see page 15) And Meaning (see page 11).
	WordMeaning (see page 16)	Looks up a localized string similar to Word (see page 15) Meaning (see page 11) Dictionary.
	WordRemovedFavouriteSuccessfully (see page 16)	Looks up a localized string similar to Word (see page 15) Has Been Removed From Favourites Successfully.
	Yellow (see page 16)	Looks up a localized string similar to Yellow.

1.1.1.1.1.1 AppResources.AppResources Constructor

C#

```
[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance",
"CA1811:AvoidUncalledPrivateCode")]
internal AppResources();
```

Description

This is AppResources, a member of class AppResources.

Body Source

```
1:
[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance",
"CA1811:AvoidUncalledPrivateCode")]
2: internal AppResources() {
3: }
```

1.1.1.1.1.2 AppResources Properties

1.1.1.1.1.2.1 AppResources.About Property

Looks up a localized string similar to About.

C#

```
public static string About;
```

1.1.1.1.1.2.2 AppResources.AboutAwesomeDictionary Property

Looks up a localized string similar to About (see page 6) Awesome Dictionary.

C#

```
public static string AboutAwesomeDictionary;
```

1.1.1.1.1.2.3 AppResources.AboutTheApp Property

Looks up a localized string similar to About (see page 6) The App (see page 23).

C#

```
public static string AboutTheApp;
```

1.1.1.1.1.2.4 AppResources.AboutTheAppText Property

Looks up a localized string similar to Hi everybody. I am with you with a new app. I investigate a lot of dictionary applications in Windows Phone and i tried to make a simple app which you will like it. If you rate the app and send your thoughts to coderserdar@outlook.com, I will be so appreciated to you. With my best regards. CoderSerdar.

C#

```
public static string AboutTheAppText;
```

1.1.1.1.1.2.5 AppResources.AddToFavourites Property

Looks up a localized string similar to Add To Favourites.

C#

```
public static string AddToFavourites;
```

1.1.1.1.1.2.6 AppResources.AtLeastOneDictionary Property

Looks up a localized string similar to You Should Select (see page 13) At Least One Dictionary.

C#

```
public static string AtLeastOneDictionary;
```

1.1.1.1.1.2.7 AppResources.Background Property

Looks up a localized string similar to Background.

C#

```
public static string Background;
```

1.1.1.1.1.2.8 AppResources.BackgroundColor Property

Looks up a localized string similar to Background (see page 6) Color.

C#

```
public static string BackgroundColor;
```

1.1.1.1.1.2.9 AppResources.BackgroundColorChangedSuccessfully Property

Looks up a localized string similar to Background (see page 6) Color Has Been Changed Successfully.

C#

```
public static string BackgroundColorChangedSuccessfully;
```

1.1.1.1.1.2.10 AppResources.BackgroundImage Property

Looks up a localized string similar to Background (see page 6) Image.

C#

```
public static string BackgroundImage;
```

1.1.1.1.1.2.11 AppResources.BackgroundImageChangedSuccessfully Property

Looks up a localized string similar to Background (see page 6) Image Has Been Changed Successfully.

C#

```
public static string BackgroundImageChangedSuccessfully;
```

1.1.1.1.1.2.12 AppResources.BackgroundImageRemovedSuccessfully Property

Looks up a localized string similar to Background (see page 6) Image Has Been Removed Successfully.

C#

```
public static string BackgroundImageRemovedSuccessfully;
```

1.1.1.1.1.2.13 AppResources.BackgroundSettingsResetSuccessfully Property

Looks up a localized string similar to Background (see page 6) Settings (see page 14) Has Been Reset Successfully.

C#

```
public static string BackgroundSettingsResetSuccessfully;
```

1.1.1.1.1.2.14 AppResources.Black Property

Looks up a localized string similar to Black.

C#

```
public static string Black;
```

1.1.1.1.1.2.15 AppResources.Blue Property

Looks up a localized string similar to Blue.

C#

```
public static string Blue;
```

1.1.1.1.1.2.16 AppResources.Brown Property

Looks up a localized string similar to Brown.

C#

```
public static string Brown;
```

1.1.1.1.1.2.17 AppResources.BuyukLugat Property

Looks up a localized string similar to Buyuk Lugat (Turkish (see page 15)->Turkish (see page 15)).

C#

```
public static string BuyukLugat;
```



1.1.1.1.1.2.18 AppResources.Cancel Property

Looks up a localized string similar to Cancel.

C#

```
public static string Cancel;
```

1.1.1.1.1.2.19 AppResources.ComputerDictionary Property

Looks up a localized string similar to Computer Dictionary ( see page 8)->Turkish ( see page 15)).

C#

```
public static string ComputerDictionary;
```

1.1.1.1.1.2.20 AppResources.ContactWithUs Property

Looks up a localized string similar to Contact With Us.

C#

```
public static string ContactWithUs;
```

1.1.1.1.1.2.21 AppResources.Culture Property

Overrides the current thread's CurrentUICulture property for all resource lookups using this strongly typed resource class.

C#

```
[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)]  
public static global::System.Globalization.CultureInfo Culture;
```

1.1.1.1.1.2.22 AppResources.DictionariesInstalledSuccessfully Property

Looks up a localized string similar to Dictionaries Have Been Installed Successfully.

C#

```
public static string DictionariesInstalledSuccessfully;
```


1.1.1.1.1.2.23 AppResources.DictionariesUninstalledSuccessfully Property

Looks up a localized string similar to Dictionaries Have Been Uninstalled Successfully.

C#

```
public static string DictionariesUninstalledSuccessfully;
```

1.1.1.1.1.2.24 AppResources.DictionaryInstall Property

Looks up a localized string similar to Dictionary Install ( see page 10).

C#

```
public static string DictionaryInstall;
```



1.1.1.1.1.2.25 AppResources.English Property

Looks up a localized string similar to English.

C#

```
public static string English;
```

1.1.1.1.1.2.26 AppResources.EnglishTurkishVol1 Property

Looks up a localized string similar to English ( see page 8)->Turkish ( see page 15) Dictionary Vol. 1.

C#

```
public static string EnglishTurkishVol1;
```

1.1.1.1.1.2.27 AppResources.EnglishTurkishVol2 Property

Looks up a localized string similar to English (see page 8)->Turkish (see page 15) Dictionary Vol. 2.

C#

```
public static string EnglishTurkishVol2;
```

1.1.1.1.1.2.28 AppResources.ExitApp Property

Looks up a localized string similar to Exit App (see page 23).

C#

```
public static string ExitApp;
```

1.1.1.1.1.2.29 AppResources.ExitAppQuestion Property

Looks up a localized string similar to Are You Sure To Exit The Application?.

C#

```
public static string ExitAppQuestion;
```

1.1.1.1.1.2.30 AppResources.FavouriteNameCount Property

Looks up a localized string similar to Favourite (see page 36) Word (see page 15) Count.

C#

```
public static string FavouriteNameCount;
```

1.1.1.1.1.2.31 AppResources.Font Property

Looks up a localized string similar to Font.

C#

```
public static string Font;
```

1.1.1.1.1.2.32 AppResources.FontFamily Property

Looks up a localized string similar to Font (see page 9) Family.

C#

```
public static string FontFamily;
```

1.1.1.1.1.2.33 AppResources.FontFamilyChangedSuccessfully Property

Looks up a localized string similar to Font (see page 9) Family Has Been Changed Successfully.

C#

```
public static string FontFamilyChangedSuccessfully;
```

1.1.1.1.1.2.34 AppResources.FontSize Property

Looks up a localized string similar to Font (see page 9) Size.

C#

```
public static string FontSize;
```

1.1.1.1.1.2.35 AppResources.FontSizeChangedSuccessfully Property

Looks up a localized string similar to Font (see page 9) Size Has Been Changed Successfully.

C#

```
public static string FontSizeChangedSuccessfully;
```

1.1.1.1.1.2.36 AppResources.GeneralSettings Property

Looks up a localized string similar to General Settings (see page 14).

C#

```
public static string GeneralSettings;
```

1.1.1.1.1.2.37 AppResources.German Property

Looks up a localized string similar to German.

C#

```
public static string German;
```

1.1.1.1.1.2.38 AppResources.GermanTurkish Property

Looks up a localized string similar to German (see page 10)->Turkish (see page 15) Dictionary.

C#

```
public static string GermanTurkish;
```

1.1.1.1.1.2.39 AppResources.Gray Property

Looks up a localized string similar to Gray.

C#

```
public static string Gray;
```

1.1.1.1.1.2.40 AppResources.Green Property

Looks up a localized string similar to Green.

C#

```
public static string Green;
```

1.1.1.1.1.2.41 AppResources.Install Property

Looks up a localized string similar to Install.

C#

```
public static string Install;
```

1.1.1.1.1.2.42 AppResources.Installing Property

Looks up a localized string similar to Installing.

C#

```
public static string Installing;
```

1.1.1.1.1.2.43 AppResources.Language Property

Looks up a localized string similar to Language.

C#

```
public static string Language;
```

1.1.1.1.1.2.44 AppResources.LanguageWarning Property

Looks up a localized string similar to You may restart the application for changes will be effect..

C#

```
public static string LanguageWarning;
```

1.1.1.1.1.2.45 AppResources.Meaning Property

Looks up a localized string similar to Meaning.

C#

```
public static string Meaning;
```

1.1.1.1.1.2.46 AppResources.MyFavourites Property

Looks up a localized string similar to My Favourites.

C#

```
public static string MyFavourites;
```

1.1.1.1.1.2.47 AppResources.Ok Property

Looks up a localized string similar to Ok.

C#

```
public static string Ok;
```



1.1.1.1.1.2.48 AppResources.Orange Property

Looks up a localized string similar to Orange.

C#

```
public static string Orange;
```

1.1.1.1.1.2.49 AppResources.OxfordDictionary Property

Looks up a localized string similar to Oxford ( see page 8)->English ( see page 8)).

C#

```
public static string OxfordDictionary;
```

1.1.1.1.1.2.50 AppResources.Purple Property

Looks up a localized string similar to Purple.

C#

```
public static string Purple;
```

1.1.1.1.1.2.51 AppResources.RandomWords Property

Looks up a localized string similar to Random 10 Words.

C#

```
public static string RandomWords;
```

1.1.1.1.1.2.52 AppResources.Rate Property

Looks up a localized string similar to Rate.

C#

```
public static string Rate;
```

1.1.1.1.1.2.53 AppResources.Red Property

Looks up a localized string similar to Red.

C#

```
public static string Red;
```

1.1.1.1.1.2.54 AppResources.RemoveBackgroundImage Property

Looks up a localized string similar to Remove Background (see page 6) Image.

C#

```
public static string RemoveBackgroundImage;
```

1.1.1.1.1.2.55 AppResources.RemoveFromFavourite Property

Looks up a localized string similar to Remove From Favourites.

C#

```
public static string RemoveFromFavourite;
```

1.1.1.1.1.2.56 AppResources.RemoveFromFavouriteQuestion Property

Looks up a localized string similar to You Will Remove The Word (see page 15) From Favourites. Are You Sure?.

C#

```
public static string RemoveFromFavouriteQuestion;
```

1.1.1.1.1.2.57 AppResources.RemoveFromFavourites Property

Looks up a localized string similar to Remove From Favourites.

C#

```
public static string RemoveFromFavourites;
```

1.1.1.1.1.2.58 AppResources.ResetSettings Property

Looks up a localized string similar to Reset Settings (see page 14).

C#

```
public static string ResetSettings;
```

1.1.1.1.1.2.59 AppResources.ResourceFlowDirection Property

Looks up a localized string similar to LeftToRight.

C#

```
public static string ResourceFlowDirection;
```

1.1.1.1.1.2.60 AppResources.ResourceLanguage Property

Looks up a localized string similar to en-US.

C#

```
public static string ResourceLanguage;
```

1.1.1.1.1.2.61 AppResources.ResourceManager Property

Returns the cached ResourceManager instance used by this class.

C#

```
[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)]  
public static global::System.Resources.ResourceManager ResourceManager;
```

1.1.1.1.1.2.62 AppResources.RisaleNur Property

Looks up a localized string similar to Risale Nur (Turkish (see page 15)->Turkish (see page 15)).

C#

```
public static string RisaleNur;
```

1.1.1.1.1.2.63 AppResources.Search Property

Looks up a localized string similar to Search.

C#

```
public static string Search;
```

1.1.1.1.1.2.64 AppResources.SearchCompleted Property

Looks up a localized string similar to Search (see page 13) Completed.

C#

```
public static string SearchCompleted;
```

1.1.1.1.1.2.65 AppResources.SearchInMeanings Property

Looks up a localized string similar to Search (see page 13) In Meanings.

C#

```
public static string SearchInMeanings;
```

1.1.1.1.1.2.66 AppResources.SearchResults Property

Looks up a localized string similar to Search (see page 13) Results.

C#

```
public static string SearchResults;
```

1.1.1.1.1.2.67 AppResources.SearchTrimFault Property

Looks up a localized string similar to Search (see page 13) Criteria Can Not Be Empty.

C#

```
public static string SearchTrimFault;
```

1.1.1.1.1.2.68 AppResources.Select Property

Looks up a localized string similar to Select.

C#

```
public static string Select;
```

1.1.1.1.1.2.69 AppResources.SelectBackgroundColor Property

Looks up a localized string similar to Select (see page 13) Background (see page 6) Color.

C#

```
public static string SelectBackgroundColor;
```

1.1.1.1.1.2.70 AppResources.Selected Property

Looks up a localized string similar to Selected.

C#

```
public static string Selected;
```

1.1.1.1.1.2.71 AppResources.SelectFontFamily Property

Looks up a localized string similar to Select (see page 13) Font (see page 9) Family.

C#

```
public static string SelectFontFamily;
```

1.1.1.1.1.2.72 AppResources.SelectFontSize Property

Looks up a localized string similar to Select (see page 13) Font (see page 9) Size.

C#

```
public static string SelectFontSize;
```

1.1.1.1.1.2.73 AppResources.SelectLanguage Property

Looks up a localized string similar to Select (see page 13) Language (see page 10).

C#

```
public static string SelectLanguage;
```

1.1.1.1.1.2.74 AppResources.SendWithAwesomeDictionaryApp Property

Looks up a localized string similar to Send With Awesome Dictionary App (see page 23).

C#

```
public static string SendWithAwesomeDictionaryApp;
```

1.1.1.1.1.2.75 AppResources.SendWithEmail Property

Looks up a localized string similar to Send With E-Mail.

C#

```
public static string SendWithEmail;
```

1.1.1.1.1.2.76 AppResources.SendWithSMS Property

Looks up a localized string similar to Send With SMS.

C#

```
public static string SendWithSMS;
```

1.1.1.1.1.2.77 AppResources.Settings Property

Looks up a localized string similar to Settings.

C#

```
public static string Settings;
```

1.1.1.1.1.2.78 AppResources.Share Property

Looks up a localized string similar to Share.

C#

```
public static string Share;
```

1.1.1.1.1.2.79 AppResources.Source Property

Looks up a localized string similar to Source.

C#

```
public static string Source;
```

1.1.1.1.1.2.80 AppResources.Statistics Property

Looks up a localized string similar to Statistics.

C#

```
public static string Statistics;
```

1.1.1.1.1.2.81 AppResources.Synchronizing Property

Looks up a localized string similar to Synchronizing.

C#

```
public static string Synchronizing;
```

1.1.1.1.1.2.82 AppResources.SystemFault Property

Looks up a localized string similar to System Has A Fault. Please Try Again Later..

C#

```
public static string SystemFault;
```

1.1.1.1.1.2.83 AppResources.TotalNameCount Property

Looks up a localized string similar to Total Word (see page 15) Count.

C#

```
public static string TotalNameCount;
```

1.1.1.1.1.2.84 AppResources.Turkish Property

Looks up a localized string similar to Turkish.

C#

```
public static string Turkish;
```

1.1.1.1.1.2.85 AppResources.Uninstall Property

Looks up a localized string similar to Uninstall.

C#

```
public static string Uninstall;
```

1.1.1.1.1.2.86 AppResources.Word Property

Looks up a localized string similar to Word.

C#

```
public static string Word;
```

1.1.1.1.1.2.87 AppResources.WordAddedFavouriteSuccessfully Property

Looks up a localized string similar to Word (see page 15) Has Been Added To Favourites Successfully.

C#

```
public static string WordAddedFavouriteSuccessfully;
```

1.1.1.1.1.2.88 AppResources.WordAlreadyFavourite Property

Looks up a localized string similar to Word (see page 15) Is Already Favourite (see page 36).

C#

```
public static string WordAlreadyFavourite;
```

1.1.1.1.1.2.89 AppResources.WordAndMeaning Property

Looks up a localized string similar to Word (see page 15) And Meaning (see page 11).

C#

```
public static string WordAndMeaning;
```

1.1.1.1.1.2.90 AppResources.WordMeaning Property

Looks up a localized string similar to Word ([see page 15](#)) Meaning ([see page 11](#)) Dictionary.

C#

```
public static string WordMeaning;
```

1.1.1.1.1.2.91 AppResources.WordRemovedFavouriteSuccessfully Property

Looks up a localized string similar to Word ([see page 15](#)) Has Been Removed From Favourites Successfully.

C#

```
public static string WordRemovedFavouriteSuccessfully;
```

1.1.1.1.1.2.92 AppResources.Yellow Property

Looks up a localized string similar to Yellow.


















C#









```
public static string Yellow;
```

1.1.2 Classes

The following table lists classes in this documentation.

Classes

	Name	Description
	AboutPage (see page 17)	This is class AwesomeDictionary>AboutPage.
	All (see page 18)	This is class AwesomeDictionary.All.
	AlmancaTurkce (see page 19)	This is class AwesomeDictionary.AlmancaTurkce.
	AlphaKeyGroup (see page 21)	This is class AwesomeDictionary.AlphaKeyGroup.
	App (see page 23)	This is class AwesomeDictionary.App.
	AppSettings (see page 25)	This is class AwesomeDictionary.AppSettings.
	AwesomeDictionaryDataContext (see page 26)	This is class AwesomeDictionary.AwesomeDictionaryDataContext.
	BackgroundColorSettingsPage (see page 29)	This is class AwesomeDictionary.BackgroundColorSettingsPage.
	BilisimSozlugu (see page 31)	This is class AwesomeDictionary.BilisimSozlugu.
	BuyukLugat (see page 33)	This is class AwesomeDictionary.BuyukLugat.
	EnglishTurkishVol1 (see page 34)	This is class AwesomeDictionary.EnglishTurkishVol1.
	EnglishTurkishVol2 (see page 35)	This is class AwesomeDictionary.EnglishTurkishVol2.
	Favourite (see page 36)	This is class AwesomeDictionary.Favourite.
	FontFamilySettingsPage (see page 37)	This is class AwesomeDictionary.FontFamilySettingsPage.
	FontSizeSettingsPage (see page 39)	This is class AwesomeDictionary.FontSizeSettingsPage.
	GeneralSettingsPage (see page 41)	This is class AwesomeDictionary.GeneralSettingsPage.
	KelimeAnlamlari (see page 46)	This is class AwesomeDictionary.KelimeAnlamlari.

	LanguageSettingsPage (see page 47)	This is class AwesomeDictionary.LanguageSettingsPage.
	LocalizedStrings (see page 49)	Provides access to string resources.
	MainPage (see page 49)	This is class AwesomeDictionary.MainPage.
	NameDetailPage (see page 50)	This is class AwesomeDictionary.NameDetailPage.
	OxfordEnglishEnglish (see page 54)	This is class AwesomeDictionary.OxfordEnglishEnglish.
	RisaleNur (see page 56)	This is class AwesomeDictionary.RisaleNur.
	SearchPage (see page 57)	This is class AwesomeDictionary.SearchPage.
	StatisticsPage (see page 58)	This is class AwesomeDictionary.StatisticsPage.

1.1.2.1 AboutPage Class

Class Hierarchy



C#

```
public class AboutPage : PhoneApplicationPage;
```

File

AboutPage.xaml.cs ([see page 60](#))

Description

This is class AwesomeDictionary.AboutPage.

Methods

	Name	Description
	AboutPage (see page 17)	This is AboutPage, a member of class AboutPage.

1.1.2.1.1 AboutPage.AboutPage Constructor

C#

```
public AboutPage();
```

Description

This is AboutPage, a member of class AboutPage.

Body Source

```

1: public AboutPage()
2: {
3:     InitializeComponent();
4:
5:     SetBackgroundColor();
6:
7:     ApplicationBar = new ApplicationBar();
8:
9:     ApplicationBarIconButton button2 = new ApplicationBarIconButton();
10:    button2.IconUri = new Uri("/Assets/SendWithMail.png", UriKind.Relative);
11:    button2.Text = AppResources.ContactWithUs;
12:    ApplicationBar.Buttons.Add(button2);
13:    button2.Click += new EventHandler(SendMailButton_Click);
14:
15:    ApplicationBarIconButton button3 = new ApplicationBarIconButton();
16:    button3.IconUri = new Uri("/Assets/Rate.png", UriKind.Relative);
17:    button3.Text = AppResources.Rate;
  
```

```
18:     ApplicationBar.Buttons.Add(button3);
19:     button3.Click += new EventHandler(RateButton_Click);
20:
21:     lblAboutTheApp.Text = AppResources.AboutTheApp;
22:     txtAbout.Text = AppResources.AboutTheAppText;
23:     txtAbout.IsEnabled = false;
24:     //var paragraph = new Paragraph();
25:     //paragraph.Inlines.Add(AppResources.AboutTheAppText);
26:     //txtAbout.Blocks.Add(paragraph);
27: }
```

1.1.2.2 All Class

Class Hierarchy

AwesomeDictionary.All

C#

```
[Index(Columns = "AllName, AllMeaning, AllNameMeaning, AllNameSource ASC", IsUnique =
false, Name = "indAllNames")]
[Table]
public class All;
```







File

All.cs (see page 63)

Description

This is class AwesomeDictionary.All.

All Properties

	Name	Description
	AllId (see page 18)	This is AllId, a member of class All.
	AllMeaning (see page 18)	This is AllMeaning, a member of class All.
	AllName (see page 19)	This is AllName, a member of class All.
	AllNameMeaning (see page 19)	This is AllNameMeaning, a member of class All.
	AllNameSource (see page 19)	This is AllNameSource, a member of class All.
	AllSource (see page 19)	This is AllSource, a member of class All.

1.1.2.2.1 All Properties

1.1.2.2.1.1 All.AllId Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int AllId;
```

Description

This is AllId, a member of class All.

1.1.2.2.1.2 All.AllMeaning Property

C#

```
[Column]
public string AllMeaning;
```

Description

This is AllMeaning, a member of class All.

1.1.2.2.1.3 All.AllName Property**C#**

```
[Column]  
public string AllName;
```

Description

This is AllName, a member of class All.

1.1.2.2.1.4 All.AllNameMeaning Property**C#**

```
[Column]  
public string AllNameMeaning;
```

Description

This is AllNameMeaning, a member of class All.

1.1.2.2.1.5 All.AllNameSource Property**C#**

```
[Column]  
public string AllNameSource;
```

Description

This is AllNameSource, a member of class All.

1.1.2.2.1.6 All.AllSource Property**C#**

```
[Column]  
public string AllSource;
```

Description

This is AllSource, a member of class All.

1.1.2.3 AlmancaTurkce Class**Class Hierarchy**

AwesomeDictionary.AlmancaTurkce

C#

```
[Table]  
public class AlmancaTurkce;
```









File

AlmancaTurkce.cs (📄 see page 64)

Description

This is class AwesomeDictionary.AlmancaTurkce.

AlmancaTurkce Properties

	Name	Description
	AlmancaTurkceld ( see page 20)	This is AlmancaTurkceld, a member of class AlmancaTurkce.
	AlmancaTurkceMeaning ( see page 20)	This is AlmancaTurkceMeaning, a member of class AlmancaTurkce.
	AlmancaTurkceName ( see page 20)	This is AlmancaTurkceName, a member of class AlmancaTurkce.
	AlmancaTurkceNameMeaning ( see page 20)	This is AlmancaTurkceNameMeaning, a member of class AlmancaTurkce.

1.1.2.3.1 AlmancaTurkce Properties**1.1.2.3.1.1 AlmancaTurkce.AlmancaTurkceld Property****C#**

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity", CanBeNull = false)]  
public int AlmancaTurkceld;
```

Description

This is AlmancaTurkceld, a member of class AlmancaTurkce.

1.1.2.3.1.2 AlmancaTurkce.AlmancaTurkceMeaning Property**C#**

```
[Column]  
public string AlmancaTurkceMeaning;
```

Description

This is AlmancaTurkceMeaning, a member of class AlmancaTurkce.

1.1.2.3.1.3 AlmancaTurkce.AlmancaTurkceName Property**C#**

```
[Column]  
public string AlmancaTurkceName;
```

Description

This is AlmancaTurkceName, a member of class AlmancaTurkce.

1.1.2.3.1.4 AlmancaTurkce.AlmancaTurkceNameMeaning Property**C#**

```
[Column]  
public string AlmancaTurkceNameMeaning;
```

Description

This is AlmancaTurkceNameMeaning, a member of class AlmancaTurkce.

1.1.2.4 AlphaKeyGroup Class

Class Hierarchy



C#

```
public class AlphaKeyGroup<T> : List<T>;
```

File

AlphaKeyGroup.cs (see page 64)

Description

This is class AwesomeDictionary.AlphaKeyGroup.

Methods

	Name	Description
	AlphaKeyGroup (see page 21)	Public constructor.

AlphaKeyGroup Delegates

Name	Description
GetKeyDelegate (see page 23)	The delegate that is used to get the key information.

AlphaKeyGroup Methods

	Name	Description
	CreateGroups (see page 21)	Create a list of AlphaGroupwith keys set by a SortedLocaleGrouping.

AlphaKeyGroup Properties

	Name	Description
	Key (see page 22)	The Key of this group.

1.1.2.4.1 AlphaKeyGroup.AlphaKeyGroup Constructor

Public constructor.

C#

```
public AlphaKeyGroup(string key) ;
```

Parameters

Parameters	Description
string key	The key for this group.

Body Source

```
1: public AlphaKeyGroup(string key)
2: {
3:     Key = key;
4: }
```

1.1.2.4.2 AlphaKeyGroup Methods

1.1.2.4.2.1 AlphaKeyGroup.CreateGroups Method

Create a list of AlphaGroupwith keys set by a SortedLocaleGrouping.

C#

```
public static List<AlphaKeyGroup<T>> CreateGroups(IEnumerable<T> items, CultureInfo ci,
    GetKeyDelegate getKey, bool sort);
```

Parameters

Parameters	Description
<code>IEnumerable<T> items</code>	The items to place in the groups.
<code>CultureInfo ci</code>	The CultureInfo to group and sort by.
<code>GetKeyDelegate getKey</code>	A delegate to get the key from an item.
<code>bool sort</code>	Will sort the data if true.

Returns

An items source for a LongListSelector

Body Source

```
1: public static List<AlphaKeyGroup<T>> CreateGroups(IEnumerable<T> items, CultureInfo ci,
    GetKeyDelegate getKey, bool sort)
2: {
3:     SortedLocaleGrouping slg = new SortedLocaleGrouping(ci);
4:     List<AlphaKeyGroup<T>> list = CreateGroups(slg);
5:
6:     foreach (T item in items)
7:     {
8:         int index = 0;
9:         if (slg.SupportsPhonetics)
10:        {
11:            //check if your database has yomi string for item
12:            //if it does not, then do you want to generate Yomi or ask the user for
13:            //index = slg.GetGroupIndex(getKey(Yomiof(item)));
14:        }
15:        else
16:        {
17:            index = slg.GetGroupIndex(getKey(item));
18:        }
19:        if (index >= 0 && index < list.Count)
20:        {
21:            list[index].Add(item);
22:        }
23:    }
24:
25:    if (sort)
26:    {
27:        foreach (AlphaKeyGroup<T> group in list)
28:        {
29:            group.Sort((c0, c1) => { return ci.CompareInfo.Compare(getKey(c0),
30:            getKey(c1)); });
31:        }
32:
33:        return list;
34:    }
```

1.1.2.4.3 AlphaKeyGroup Properties**1.1.2.4.3.1 AlphaKeyGroup.Key Property**

The Key of this group.

C#

```
public string Key;
```

1.1.2.4.4 AlphaKeyGroup Delegates

1.1.2.4.4.1 AlphaKeyGroup.GetKeyDelegate Delegate

The delegate that is used to get the key information.

C#

```
public delegate string GetKeyDelegate(T item);
```

Parameters

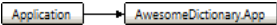
Parameters	Description
item	An object of type T

Returns

The key value to use for this object

1.1.2.5 App Class

Class Hierarchy



C#

```
public class App : Application;
```

File

App.xaml.cs ([see page 66](#))

Description

This is class AwesomeDictionary.App.

Methods

	Name	Description
	App (see page 23)	Constructor for the Application object.

App Methods

	Name	Description
	ReadFile (see page 24)	This is ReadFile, a member of class App.

App Properties

	Name	Description
	RootFrame (see page 25)	Provides easy access to the root frame of the Phone Application.

1.1.2.5.1 App.App Constructor

Constructor for the Application object.

C#

```
public App();
```

Body Source

```
1: public App()
2: {
```

```

3:      // Global handler for uncaught exceptions.
4:      UnhandledException += Application_UnhandledException;
5:
6:      // Standard XAML initialization
7:      InitializeComponent();
8:
9:      // ayarlardan teması açık renk bile olsa
10:     // kapalı gibi çalışmasını sağlayacak bir nuget paketi yüklendi
11:     // bu sorunu gideriyor
12:     ThemeManager.ToDarkTheme();
13:
14:     // Phone-specific initialization
15:     InitializePhoneApplication();
16:
17:     // Language display initialization
18:     InitializeLanguage();
19:
20:     // Show graphics profiling information while debugging.
21:     if (Debugger.IsAttached)
22:     {
23:         // Display the current frame rate counters.
24:         Application.Current.Host.Settings.EnableFrameRateCounter = true;
25:
26:         // Show the areas of the app that are being redrawn in each frame.
27:         //Application.Current.Host.Settings.EnableRedrawRegions = true;
28:
29:         // Enable non-production analysis visualization mode,
30:         // which shows areas of a page that are handed off to GPU with a colored
31:         // overlay.
32:         //Application.Current.Host.Settings.EnableCacheVisualization = true;
33:
34:         // Prevent the screen from turning off while under the debugger by disabling
35:         // the application's idle detection.
36:         // Caution:- Use this under debug mode only. Application that disables user
37:         // idle detection will continue to run
38:         // and consume battery power when the user is not using the phone.
39:         PhoneApplicationService.Current.UserIdleDetectionMode =
40:         IdleDetectionMode.Disabled;
41:     }
42: }

```

1.1.2.5.2 App Methods

1.1.2.5.2.1 App.ReadFile Method

C#

```
public string ReadFile(string filePath);
```

Description

This is ReadFile, a member of class App.

Body Source

```

1: public string ReadFile(string filePath)
2: {
3:     var ResrouceStream = Application.GetResourceStream(new Uri(filePath,
4: UriKind.Relative));
5:     if (ResrouceStream != null)
6:     {
7:         Stream myFileStream = ResrouceStream.Stream;
8:         if (myFileStream.CanRead)
9:         {
10:             StreamReader myStreamReader = new StreamReader(myFileStream);
11:             return myStreamReader.ReadToEnd();
12:         }
13:     }
14: }

```

```
12:         }  
13:     }  
14:     return "";  
15: }
```

1.1.2.5.3 App Properties

1.1.2.5.3.1 App.RootFrame Property

Provides easy access to the root frame of the Phone Application.

C#

```
public static PhoneApplicationFrame RootFrame;
```

Returns

The root frame of the Phone Application.

1.1.2.6 AppSettings Class

Class Hierarchy



C#

```
[Table]  
public class AppSettings;
```







File

AppSettings.cs ([see page 87](#))

Description

This is class AwesomeDictionary.AppSettings.

AppSettings Properties

	Name	Description
	AppBackgroundColor (see page 25)	This is AppBackgroundColor, a member of class AppSettings.
	AppBackgroundImage (see page 26)	This is AppBackgroundImage, a member of class AppSettings.
	AppLangName (see page 26)	This is AppLangName, a member of class AppSettings.
	AppSettingsId (see page 26)	This is AppSettingsId, a member of class AppSettings.
	FontFamily (see page 26)	This is FontFamily, a member of class AppSettings.
	FontSize (see page 26)	This is FontSize, a member of class AppSettings.

1.1.2.6.1 AppSettings Properties

1.1.2.6.1.1 AppSettings.AppBackgroundColor Property

C#

```
[Column]  
public string AppBackgroundColor;
```

Description

This is AppBackgroundColor, a member of class AppSettings.

1.1.2.6.1.2 AppSettings.AppBackgroundImage Property

C#

```
[Column(DbType = "Image", UpdateCheck = UpdateCheck.Never)]  
public byte AppBackgroundImage;
```

Description

This is AppBackgroundImage, a member of class AppSettings.

1.1.2.6.1.3 AppSettings.AppLangName Property

C#

```
[Column]  
public string AppLangName;
```

Description

This is AppLangName, a member of class AppSettings.

1.1.2.6.1.4 AppSettings.AppSettingsId Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",  
CanBeNull = false)]  
public int AppSettingsId;
```

Description

This is AppSettingsId, a member of class AppSettings.

1.1.2.6.1.5 AppSettings.FontFamily Property

C#

```
[Column]  
public string FontFamily;
```

Description

This is FontFamily, a member of class AppSettings.

1.1.2.6.1.6 AppSettings.FontSize Property

C#

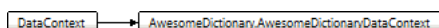
```
[Column]  
public string FontSize;
```

Description

This is FontSize, a member of class AppSettings.

1.1.2.7 AwesomeDictionaryDataContext Class

Class Hierarchy



C#

```
public class AwesomeDictionaryDataContext : DataContext;
```


File

AwesomeDictionaryDataContext.cs ([🔗](#) see page 88)













Description

This is class AwesomeDictionary.AwesomeDictionaryDataContext.

Methods

	Name	Description
	AwesomeDictionaryDataContext (🔗 see page 27)	This is AwesomeDictionaryDataContext, a member of class AwesomeDictionaryDataContext.

AwesomeDictionaryDataContext Fields

	Name	Description
	AllNames (🔗 see page 27)	This is AllNames, a member of class AwesomeDictionaryDataContext.
	AlmancaTurkces (🔗 see page 28)	This is AlmancaTurkces, a member of class AwesomeDictionaryDataContext.
	AppSettings (🔗 see page 28)	This is AppSettings, a member of class AwesomeDictionaryDataContext.
	Bilisims (🔗 see page 28)	This is Bilisims, a member of class AwesomeDictionaryDataContext.
	BuyukLugats (🔗 see page 28)	This is BuyukLugats, a member of class AwesomeDictionaryDataContext.
	ConnectionString (🔗 see page 28)	This is ConnectionString, a member of class AwesomeDictionaryDataContext.
	EnglishTurkishVol1s (🔗 see page 28)	This is EnglishTurkishVol1s, a member of class AwesomeDictionaryDataContext.
	EnglishTurkishVol2s (🔗 see page 28)	This is EnglishTurkishVol2s, a member of class AwesomeDictionaryDataContext.
	Favourites (🔗 see page 29)	This is Favourites, a member of class AwesomeDictionaryDataContext.
	Kelimes (🔗 see page 29)	This is Kelimes, a member of class AwesomeDictionaryDataContext.
	Oxfords (🔗 see page 29)	This is Oxfords, a member of class AwesomeDictionaryDataContext.
	RisaleNurs (🔗 see page 29)	This is RisaleNurs, a member of class AwesomeDictionaryDataContext.

1.1.2.7.1 AwesomeDictionaryDataContext.AwesomeDictionaryDataContext Constructor

C#

```
public AwesomeDictionaryDataContext(string connectionString);
```

Description

This is AwesomeDictionaryDataContext, a member of class AwesomeDictionaryDataContext.

Body Source

```
1: public AwesomeDictionaryDataContext(string connectionString)
2: : base(connectionString) { }
```

1.1.2.7.2 AwesomeDictionaryDataContext Fields

1.1.2.7.2.1 AwesomeDictionaryDataContext.AllNames Field

C#

```
public Table<All> AllNames;
```

Description

This is AllNames, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.2 AwesomeDictionaryDataContext.AlmancaTurkces Field

C#

```
public Table<AlmancaTurkce> AlmancaTurkces;
```

Description

This is AlmancaTurkces, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.3 AwesomeDictionaryDataContext.AppSettings Field

C#

```
public Table<AppSettings> AppSettings;
```

Description

This is AppSettings, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.4 AwesomeDictionaryDataContext.Bilisims Field

C#

```
public Table<BilisimSozlugu> Bilisims;
```

Description

This is Bilisims, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.5 AwesomeDictionaryDataContext.BuyukLugats Field

C#

```
public Table<BuyukLugat> BuyukLugats;
```

Description

This is BuyukLugats, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.6 AwesomeDictionaryDataContext.ConnectionString Field

C#

```
public const string ConnectionString = @"Data Source=isostore:/AwesomeDictionary.sdf; Max Database Size=256; Max Buffer Size=4096;";
```

Description

This is ConnectionString, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.7 AwesomeDictionaryDataContext.EnglishTurkishVol1s Field

C#

```
public Table<EnglishTurkishVoll> EnglishTurkishVolls;
```

Description

This is EnglishTurkishVol1s, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.8 AwesomeDictionaryDataContext.EnglishTurkishVol2s Field

C#

```
public Table<EnglishTurkishVol2> EnglishTurkishVol2s;
```


Description

This is EnglishTurkishVol2s, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.9 AwesomeDictionaryDataContext.Favourites Field**C#**

```
public Table<Favourite> Favourites;
```

Description

This is Favourites, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.10 AwesomeDictionaryDataContext.Kelimes Field**C#**

```
public Table<KelimeAnlamlari> Kelimes;
```

Description

This is Kelimes, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.11 AwesomeDictionaryDataContext.Oxfords Field**C#**

```
public Table<OxfordEnglishEnglish> Oxfords;
```

Description

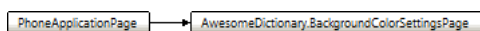
This is Oxfords, a member of class AwesomeDictionaryDataContext.

1.1.2.7.2.12 AwesomeDictionaryDataContext.RisaleNurs Field**C#**

```
public Table<RisaleNur> RisaleNurs;
```

Description

This is RisaleNurs, a member of class AwesomeDictionaryDataContext.

1.1.2.8 BackgroundColorSettingsPage Class**Class Hierarchy****C#**

```
public class BackgroundColorSettingsPage : PhoneApplicationPage;
```


File

BackgroundColorSettingsPage.xaml.cs (📄 see page 89)




Description

This is class AwesomeDictionary.BackgroundColorSettingsPage.

Methods

	Name	Description
	BackgroundColorSettingsPage (🔗 see page 30)	This is BackgroundColorSettingsPage, a member of class BackgroundColorSettingsPage.

BackgroundColorSettingsPage Methods

	Name	Description
	OnFragmentNavigation (🔗 see page 30)	This is OnFragmentNavigation, a member of class BackgroundColorSettingsPage.
	OnNavigatedFrom (🔗 see page 31)	This is OnNavigatedFrom, a member of class BackgroundColorSettingsPage.
	OnNavigatedTo (🔗 see page 31)	This is OnNavigatedTo, a member of class BackgroundColorSettingsPage.

1.1.2.8.1 BackgroundColorSettingsPage.BackgroundColorSettingsPage Constructor**C#**

```
public BackgroundColorSettingsPage();
```

Description

This is BackgroundColorSettingsPage, a member of class BackgroundColorSettingsPage.

Body Source

```
1: public BackgroundColorSettingsPage()
2: {
3:     InitializeComponent();
4:
5:     lstBackgroundColor.Items.Clear();
6:     lstBackgroundColor.Items.Add(AppResources.Black);
7:     lstBackgroundColor.Items.Add(AppResources.Blue);
8:     lstBackgroundColor.Items.Add(AppResources.Brown);
9:     lstBackgroundColor.Items.Add(AppResources.Gray);
10:    lstBackgroundColor.Items.Add(AppResources.Green);
11:    lstBackgroundColor.Items.Add(AppResources.Orange);
12:    lstBackgroundColor.Items.Add(AppResources.Purple);
13:    lstBackgroundColor.Items.Add(AppResources.Red);
14:    lstBackgroundColor.Items.Add(AppResources.Yellow);
15:    lstBackgroundColor.SelectedIndex = -1;
16:
17:    lblBackgroundColor.Text = AppResources.SelectBackgroundColor;
18:    lblGeneralSettings.Text = AppResources.GeneralSettings;
19:
20:    SetBackgroundColor();
21: }
```

1.1.2.8.2 BackgroundColorSettingsPage Methods**1.1.2.8.2.1 BackgroundColorSettingsPage.OnFragmentNavigation Method****C#**

```
protected override void OnFragmentNavigation(FragmentNavigationEventArgs e);
```

Description

This is OnFragmentNavigation, a member of class BackgroundColorSettingsPage.

Body Source

```

1: protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
2: {
3:     // displays "Fragment: Detail"
4:     //MessageBox.Show("Folder Id: " + e.Fragment);
5:     // base.OnFragmentNavigation(e);
6:     // using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
7:     // {
8:     //     lblGeneralSettings.Text = AppResources.GeneralSettings;
9:     //     lblBackgroundColor.Text = AppResources.SelectFontSize;
10:    // }
11: }

```

1.1.2.8.2.2 BackgroundColorSettingsPage.OnNavigatedFrom Method**C#**

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class BackgroundColorSettingsPage.

Body Source

```

1: protected override void OnNavigatedFrom(NavigationEventArgs e)
2: {
3:     base.OnNavigatedFrom(e);
4:     //while (NavigationService.CanGoBack)
5:     //NavigationService.RemoveBackEntry();
6:
7: }

```

1.1.2.8.2.3 BackgroundColorSettingsPage.OnNavigatedTo Method**C#**

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

Description

This is OnNavigatedTo, a member of class BackgroundColorSettingsPage.

Body Source

```

1: protected override void OnNavigatedTo(NavigationEventArgs e)
2: {
3:     base.OnNavigatedTo(e);
4:     //SetBackgroundColor();
5:     //while (NavigationService.CanGoBack)
6:     //NavigationService.RemoveBackEntry();
7:
8: }

```

1.1.2.9 BilisimSozlugu Class**Class Hierarchy**

```
AwesomeDictionary.BilisimSozlugu
```

C#

```
[Table]
public class BilisimSozlugu;
```









File

BilisimSozlugu.cs (📄 see page 92)

Description

This is class AwesomeDictionary.BilisimSozlugu.

BilisimSozlugu Properties

	Name	Description
	BilisimSozluguld ( see page 32)	This is BilisimSozluguld, a member of class BilisimSozlugu.
	BilisimSozluguMeaning ( see page 32)	This is BilisimSozluguMeaning, a member of class BilisimSozlugu.
	BilisimSozluguName ( see page 32)	This is BilisimSozluguName, a member of class BilisimSozlugu.
	BilisimSozluguNameMeaning ( see page 32)	This is BilisimSozluguNameMeaning, a member of class BilisimSozlugu.

1.1.2.9.1 BilisimSozlugu Properties

1.1.2.9.1.1 BilisimSozlugu.BilisimSozluguld Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",  
CanBeNull = false)]  
public int BilisimSozluguId;
```

Description

This is BilisimSozluguld, a member of class BilisimSozlugu.

1.1.2.9.1.2 BilisimSozlugu.BilisimSozluguMeaning Property

C#

```
[Column]  
public string BilisimSozluguMeaning;
```

Description

This is BilisimSozluguMeaning, a member of class BilisimSozlugu.

1.1.2.9.1.3 BilisimSozlugu.BilisimSozluguName Property

C#

```
[Column]  
public string BilisimSozluguName;
```

Description

This is BilisimSozluguName, a member of class BilisimSozlugu.

1.1.2.9.1.4 BilisimSozlugu.BilisimSozluguNameMeaning Property

C#

```
[Column]  
public string BilisimSozluguNameMeaning;
```

Description

This is BilisimSozluguNameMeaning, a member of class BilisimSozlugu.

1.1.2.10 BuyukLugat Class

Class Hierarchy

AwesomeDictionary.BuyukLugat

C#

```
[Table]
public class BuyukLugat;
```





File

BuyukLugat.cs (see page 93)

Description

This is class AwesomeDictionary.BuyukLugat.

BuyukLugat Properties

	Name	Description
	BuyukLugatId (see page 33)	This is BuyukLugatId, a member of class BuyukLugat.
	BuyukLugatMeaning (see page 33)	This is BuyukLugatMeaning, a member of class BuyukLugat.
	BuyukLugatName (see page 33)	This is BuyukLugatName, a member of class BuyukLugat.
	BuyukLugatNameMeaning (see page 34)	This is BuyukLugatNameMeaning, a member of class BuyukLugat.

1.1.2.10.1 BuyukLugat Properties

1.1.2.10.1.1 BuyukLugat.BuyukLugatId Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int BuyukLugatId;
```

Description

This is BuyukLugatId, a member of class BuyukLugat.

1.1.2.10.1.2 BuyukLugat.BuyukLugatMeaning Property

C#

```
[Column]
public string BuyukLugatMeaning;
```

Description

This is BuyukLugatMeaning, a member of class BuyukLugat.

1.1.2.10.1.3 BuyukLugat.BuyukLugatName Property

C#

```
[Column]
public string BuyukLugatName;
```

Description

This is BuyukLugatName, a member of class BuyukLugat.

1.1.2.10.1.4 BuyukLugat.BuyukLugatNameMeaning Property**C#**

```
[Column]
public string BuyukLugatNameMeaning;
```

Description

This is BuyukLugatNameMeaning, a member of class BuyukLugat.

1.1.2.11 EnglishTurkishVol1 Class**Class Hierarchy**

```
AwesomeDictionary.EnglishTurkishVol1
```

C#

```
[Table]
public class EnglishTurkishVol1;
```





File

EnglishTurkishVol1.cs (see page 94)

Description

This is class AwesomeDictionary.EnglishTurkishVol1.

EnglishTurkishVol1 Properties

	Name	Description
	EnglishVol1Id (see page 34)	This is EnglishVol1Id, a member of class EnglishTurkishVol1.
	EnglishVol1Meaning (see page 34)	This is EnglishVol1Meaning, a member of class EnglishTurkishVol1.
	EnglishVol1Name (see page 35)	This is EnglishVol1Name, a member of class EnglishTurkishVol1.
	EnglishVol1NameMeaning (see page 35)	This is EnglishVol1NameMeaning, a member of class EnglishTurkishVol1.

1.1.2.11.1 EnglishTurkishVol1 Properties**1.1.2.11.1.1 EnglishTurkishVol1.EnglishVol1Id Property****C#**

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int EnglishVol1Id;
```

Description

This is EnglishVol1Id, a member of class EnglishTurkishVol1.

1.1.2.11.1.2 EnglishTurkishVol1.EnglishVol1Meaning Property**C#**

```
[Column]
```

```
public string EnglishVol1Meaning;
```

Description

This is EnglishVol1Meaning, a member of class EnglishTurkishVol1.

1.1.2.11.1.3 EnglishTurkishVol1.EnglishVol1Name Property

C#

```
[Column]  
public string EnglishVol1Name;
```

Description

This is EnglishVol1Name, a member of class EnglishTurkishVol1.

1.1.2.11.1.4 EnglishTurkishVol1.EnglishVol1NameMeaning Property

C#

```
[Column]  
public string EnglishVol1NameMeaning;
```

Description

This is EnglishVol1NameMeaning, a member of class EnglishTurkishVol1.

1.1.2.12 EnglishTurkishVol2 Class

Class Hierarchy

AwesomeDictionary.EnglishTurkishVol2

C#

```
[Table]  
public class EnglishTurkishVol2;
```





File

EnglishTurkishVol2.cs (see page 94)

Description

This is class AwesomeDictionary.EnglishTurkishVol2.

EnglishTurkishVol2 Properties

	Name	Description
	EnglishVol2Id (see page 35)	This is EnglishVol2Id, a member of class EnglishTurkishVol2.
	EnglishVol2Meaning (see page 36)	This is EnglishVol2Meaning, a member of class EnglishTurkishVol2.
	EnglishVol2Name (see page 36)	This is EnglishVol2Name, a member of class EnglishTurkishVol2.
	EnglishVol2NameMeaning (see page 36)	This is EnglishVol2NameMeaning, a member of class EnglishTurkishVol2.

1.1.2.12.1 EnglishTurkishVol2 Properties

1.1.2.12.1.1 EnglishTurkishVol2.EnglishVol2Id Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
```

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.

```
CanBeNull = false)]  
public int EnglishVol2Id;
```

Description

This is EnglishVol2Id, a member of class EnglishTurkishVol2.

1.1.2.12.1.2 EnglishTurkishVol2.EnglishVol2Meaning Property**C#**

```
[Column]  
public string EnglishVol2Meaning;
```

Description

This is EnglishVol2Meaning, a member of class EnglishTurkishVol2.

1.1.2.12.1.3 EnglishTurkishVol2.EnglishVol2Name Property**C#**

```
[Column]  
public string EnglishVol2Name;
```

Description

This is EnglishVol2Name, a member of class EnglishTurkishVol2.

1.1.2.12.1.4 EnglishTurkishVol2.EnglishVol2NameMeaning Property**C#**

```
[Column]  
public string EnglishVol2NameMeaning;
```

Description

This is EnglishVol2NameMeaning, a member of class EnglishTurkishVol2.

1.1.2.13 Favourite Class**Class Hierarchy**

AwesomeDictionary.Favourite

C#

```
[Table]  
public class Favourite;
```




File

Favourite.cs ([see page 95](#))

Description

This is class AwesomeDictionary.Favourite.

Favourite Properties

	Name	Description
	FavouriteAllId (see page 37)	This is FavouriteAllId, a member of class Favourite.
	FavouriteId (see page 37)	This is FavouriteId, a member of class Favourite.
	FavouriteName (see page 37)	This is FavouriteName, a member of class Favourite.

1.1.2.13.1 Favourite Properties

1.1.2.13.1.1 Favourite.FavouriteAllId Property

C#

```
[Column]
public int FavouriteAllId;
```

Description

This is FavouriteAllId, a member of class Favourite.

1.1.2.13.1.2 Favourite.FavouriteId Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int FavouriteId;
```

Description

This is FavouriteId, a member of class Favourite.

1.1.2.13.1.3 Favourite.FavouriteName Property

C#

```
[Column]
public string FavouriteName;
```

Description

This is FavouriteName, a member of class Favourite.

1.1.2.14 FontFamilySettingsPage Class



C#

```
public class FontFamilySettingsPage : PhoneApplicationPage;
```

File

FontFamilySettingsPage.xaml.cs (🔗 see page 96)

Description

This is class AwesomeDictionary.FontFamilySettingsPage.

Methods

	Name	Description
🔗	FontFamilySettingsPage (🔗 see page 38)	This is FontFamilySettingsPage, a member of class FontFamilySettingsPage.

FontFamilySettingsPage Methods

	Name	Description
☞	OnFragmentNavigation (☞ see page 38)	This is OnFragmentNavigation, a member of class FontFamilySettingsPage.
☞	OnNavigatedFrom (☞ see page 39)	This is OnNavigatedFrom, a member of class FontFamilySettingsPage.
☞	OnNavigatedTo (☞ see page 39)	This is OnNavigatedTo, a member of class FontFamilySettingsPage.

1.1.2.14.1 FontFamilySettingsPage.FontFamilySettingsPage Constructor**C#**

```
public FontFamilySettingsPage();
```

Description

This is FontFamilySettingsPage, a member of class FontFamilySettingsPage.

Body Source

```
1: public FontFamilySettingsPage()
2: {
3:     InitializeComponent();
4:
5:     lstFontFamily.Items.Clear();
6:     lstFontFamily.Items.Add("Arial");
7:     lstFontFamily.Items.Add("Arial Black");
8:     lstFontFamily.Items.Add("Baskerville Old Face");
9:     lstFontFamily.Items.Add("Berlin Sans FB");
10:    lstFontFamily.Items.Add("Albumman Old Style");
11:    lstFontFamily.Items.Add("Calibri");
12:    lstFontFamily.Items.Add("Cambria");
13:    lstFontFamily.Items.Add("Candara");
14:    lstFontFamily.Items.Add("Comic Sans MS");
15:    lstFontFamily.Items.Add("Consolas");
16:    lstFontFamily.Items.Add("Constantia");
17:    lstFontFamily.Items.Add("Courier New");
18:    lstFontFamily.Items.Add("DokChampa");
19:    lstFontFamily.Items.Add("Ebrima");
20:    lstFontFamily.Items.Add("Georgia");
21:    lstFontFamily.Items.Add("Lucida Sans Unicode");
22:    lstFontFamily.Items.Add("Meiryo UI");
23:    lstFontFamily.Items.Add("Microsoft YaHei");
24:    lstFontFamily.Items.Add("Malgun Gothic");
25:    lstFontFamily.Items.Add("Segoe UI");
26:    lstFontFamily.Items.Add("Segoe WP");
27:    lstFontFamily.Items.Add("Tahoma");
28:    lstFontFamily.Items.Add("Trebuchet MS");
29:    lstFontFamily.Items.Add("Times New Roman");
30:    lstFontFamily.Items.Add("Verdana");
31:    lstFontFamily.SelectedIndex = -1;
32:
33:    lblFontFamily.Text = AppResources.SelectFontFamily;
34:    lblGeneralSettings.Text = AppResources.GeneralSettings;
35:
36:    SetBackgroundColor();
37: }
```

1.1.2.14.2 FontFamilySettingsPage Methods**1.1.2.14.2.1 FontFamilySettingsPage.OnFragmentNavigation Method****C#**

```
protected override void OnFragmentNavigation(FragmentNavigationEventArgs e);
```

Description

This is OnFragmentNavigation, a member of class FontFamilySettingsPage.

Body Source

```

1: protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
2: {
3:     // displays "Fragment: Detail"
4:     //MessageBox.Show("Folder Id: " + e.Fragment);
5:     base.OnFragmentNavigation(e);
6:     //artistId = int.Parse(e.Fragment);
7:     //using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
8:     //{
9:     //    var artist = context.Artists.Where(j => j.ArtistId.Equals(artistId)).Single()
as Artist;
10:    //    lblArtistName.Text = artist.ArtistName;
11:    //    lblFontFamily.Text = AppResources.SelectFontFamily;
12:    //}
13:
14: }
```

1.1.2.14.2.2 FontFamilySettingsPage.OnNavigatedFrom Method**C#**

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class FontFamilySettingsPage.

Body Source

```

1: protected override void OnNavigatedFrom(NavigationEventArgs e)
2: {
3:     base.OnNavigatedFrom(e);
4: }
```

1.1.2.14.2.3 FontFamilySettingsPage.OnNavigatedTo Method**C#**

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

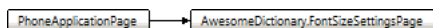
Description

This is OnNavigatedTo, a member of class FontFamilySettingsPage.

Body Source

```

1: protected override void OnNavigatedTo(NavigationEventArgs e)
2: {
3:     base.OnNavigatedTo(e);
4: }
```

1.1.2.15 FontSizeSettingsPage Class**Class Hierarchy****C#**

```
public class FontSizeSettingsPage : PhoneApplicationPage;
```

File

FontSizeSettingsPage.xaml.cs (see page 98)

Description

This is class AwesomeDictionary.FontSizeSettingsPage.

Methods

	Name	Description
🔗	FontSizeSettingsPage (🔗 see page 40)	This is FontSizeSettingsPage, a member of class FontSizeSettingsPage.

FontSizeSettingsPage Methods

	Name	Description
🔗	OnFragmentNavigation (🔗 see page 40)	This is OnFragmentNavigation, a member of class FontSizeSettingsPage.
🔗	OnNavigatedFrom (🔗 see page 41)	This is OnNavigatedFrom, a member of class FontSizeSettingsPage.
🔗	OnNavigatedTo (🔗 see page 41)	This is OnNavigatedTo, a member of class FontSizeSettingsPage.

1.1.2.15.1 FontSizeSettingsPage.FontSizeSettingsPage Constructor

C#

```
public FontSizeSettingsPage();
```

Description

This is FontSizeSettingsPage, a member of class FontSizeSettingsPage.

Body Source

```
1: public FontSizeSettingsPage()  
2: {  
3:     InitializeComponent();  
4:  
5:     lstFontSize.Items.Clear();  
6:     lstFontSize.Items.Add("14");  
7:     lstFontSize.Items.Add("18");  
8:     lstFontSize.Items.Add("22");  
9:     lstFontSize.Items.Add("26");  
10:    lstFontSize.Items.Add("28");  
11:    lstFontSize.Items.Add("30");  
12:    lstFontSize.Items.Add("32");  
13:    lstFontSize.Items.Add("34");  
14:    lstFontSize.Items.Add("36");  
15:    lstFontSize.Items.Add("38");  
16:    lstFontSize.Items.Add("40");  
17:    lstFontSize.Items.Add("42");  
18:    lstFontSize.Items.Add("44");  
19:    lstFontSize.Items.Add("64");  
20:    lstFontSize.Items.Add("72");  
21:    lstFontSize.SelectedIndex = -1;  
22:  
23:    lblGeneralSettings.Text = AppResources.GeneralSettings;  
24:    lblFontSize.Text = AppResources.SelectFontSize;  
25:  
26:    SetBackgroundColor();  
27: }
```

1.1.2.15.2 FontSizeSettingsPage Methods

1.1.2.15.2.1 FontSizeSettingsPage.OnFragmentNavigation Method

C#

```
protected override void OnFragmentNavigation(FragmentNavigationEventArgs e);
```

Description

This is OnFragmentNavigation, a member of class FontSizeSettingsPage.

Body Source

```

1: protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
2: {
3:     // displays "Fragment: Detail"
4:     //MessageBox.Show("Folder Id: " + e.Fragment);
5:     base.OnFragmentNavigation(e);
6:     //artistId = int.Parse(e.Fragment);
7:     //using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
8:     //{
9:     //    var artist = context.Artists.Where(j => j.ArtistId.Equals(artistId)).Single()
as Artist;
10:    //    lblArtistName.Text = artist.ArtistName;
11:    //    lblFontSize.Text = AppResources.SelectFontSize;
12:    //}
13:
14: }
```

1.1.2.15.2.2 FontSizeSettingsPage.OnNavigatedFrom Method**C#**

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class FontSizeSettingsPage.

Body Source

```

1: protected override void OnNavigatedFrom(NavigationEventArgs e)
2: {
3:     base.OnNavigatedFrom(e);
4: }
```

1.1.2.15.2.3 FontSizeSettingsPage.OnNavigatedTo Method**C#**

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

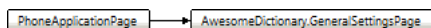
Description

This is OnNavigatedTo, a member of class FontSizeSettingsPage.

Body Source

```

1: protected override void OnNavigatedTo(NavigationEventArgs e)
2: {
3:     base.OnNavigatedTo(e);
4: }
```

1.1.2.16 GeneralSettingsPage Class**Class Hierarchy****C#**

```
public class GeneralSettingsPage : PhoneApplicationPage;
```

File

GeneralSettingsPage.xaml.cs (see page 101)

Description

This is class AwesomeDictionary.GeneralSettingsPage.

Methods

	Name	Description
🔗	GeneralSettingsPage (🔗 see page 42)	This is GeneralSettingsPage, a member of class GeneralSettingsPage.

GeneralSettingsPage Methods

	Name	Description
🔗	OnNavigatedFrom (🔗 see page 45)	This is OnNavigatedFrom, a member of class GeneralSettingsPage.
🔗	OnNavigatedTo (🔗 see page 45)	This is OnNavigatedTo, a member of class GeneralSettingsPage.
🔗	ReadFile (🔗 see page 45)	This is ReadFile, a member of class GeneralSettingsPage.

1.1.2.16.1 GeneralSettingsPage.GeneralSettingsPage Constructor**C#**

```
public GeneralSettingsPage();
```

Description

This is GeneralSettingsPage, a member of class GeneralSettingsPage.

Body Source

```
1: public GeneralSettingsPage()
2: {
3:     InitializeComponent();
4:     SetBackgroundColor();
5:
6:     pvGeneralSettings.Title = AppResources.GeneralSettings;
7:
8:     piLanguage.Header = AppResources.Language;
9:     piDictionaryInstall.Header = AppResources.DictionaryInstall;
10:    //piOtherSettings.Header = AppResources.OtherSettings;
11:    piBackground.Header = AppResources.Background;
12:
13:    //lblOneDrive.Text = AppResources.OneDrive;
14:
15:    btnLanguage.Content = AppResources.Select;
16:    btnBackgroundColor.Content = AppResources.Select;
17:    //btnOneDrive.Content = AppResources.Login;
18:    //btnOneDrive.SignInText = AppResources.SignIn;
19:    //btnOneDrive.SignOutText = AppResources.SignOut;
20:    txtInstalling.Text = AppResources.Installing;
21:
22:    pbInstall.Visibility = Visibility.Collapsed;
23:    txtInstalling.Visibility = Visibility.Collapsed;
24:    txtInstalling.BorderBrush = this.LayoutRoot.Background;
25:
26:    btnRemoveBackgroundImage.Content = AppResources.RemoveBackgroundImage;
27:    lblBackgroundImage.Text = AppResources.BackgroundImage;
28:    btnBackgroundImage.Content = AppResources.Select;
29:    btnResetSettings.Content = AppResources.ResetSettings;
30:
31:    piFont.Header = AppResources.Font;
32:    btnFontFamily.Content = AppResources.Select;
33:    btnFontSize.Content = AppResources.Select;
34:
35:    btnInstall.Content = AppResources.Install;
36:    btnUninstall.Content = AppResources.Uninstall;
37:
38:    txtBuyukLugat.Text = AppResources.BuyukLugat + " (49331 " + AppResources.Word +
```

```

");
39:         txtComputer.Text = AppResources.ComputerDictionary + " (3508 " + AppResources.Word
+ " )";
40:         txtGerman.Text = AppResources.GermanTurkish + " (17526 " + AppResources.Word + " )";
41:         txtOxford.Text = AppResources.OxfordDictionary + " (36369 " + AppResources.Word +
+ " )";
42:         txtWordMeaning.Text = AppResources.WordMeaning + " (10535 " + AppResources.Word +
+ " )";
43:         txtRisaleNur.Text = AppResources.RisaleNur + " (9478 " + AppResources.Word + " )";
44:         txtEnglishVol1.Text = AppResources.EnglishTurkishVol1 + " (127157 " +
AppResources.Word + " )";
45:         txtEnglishVol2.Text = AppResources.EnglishTurkishVol2 + " (3699 " +
AppResources.Word + " )";
46:
47:
48:         //cbSync.Content = AppResources.SyncOnOneFile;
49:         //cbSync.IsEnabled = false;
50:
51:         SetBackgroundColor();
52:
53:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
54:         {
55:             var appSettings = context.AppSettings.First() as AppSettings;
56:
57:             lblFontFamily.Text = AppResources.FontFamily + " (" + AppResources.Selected +
+ " : " + appSettings.FontFamily + " )";
58:             lblFontSize.Text = AppResources.FontSize + " (" + AppResources.Selected + " : "
+ appSettings.FontSize + " )";
59:
60:             if (appSettings.AppLangName == "EN")
61:             {
62:                 lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected +
+ " : " + AppResources.English + " )";
63:             }
64:             if (appSettings.AppLangName == "TR")
65:             {
66:                 lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected +
+ " : " + AppResources.Turkish + " )";
67:             }
68:             if (appSettings.AppLangName == "DE")
69:             {
70:                 lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected +
+ " : " + AppResources.German + " )";
71:             }
72:             //if (appSettings.AppLangName == "ES")
73:             //{
74:             //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Spanish + " )";
75:             //}
76:
77:             //if (appSettings.AppLangName == "PT")
78:             //{
79:             //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Portuguese + " )";
80:             //}
81:             //if (appSettings.AppLangName == "AR")
82:             //{
83:             //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Arabic + " )";
84:             //}
85:             //if (appSettings.AppLangName == "FA")
86:             //{
87:             //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Persian + " )";
88:             //}
89:             //if (appSettings.AppLangName == "IT")
90:             //{
91:             //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected

```

```

+ " : " + AppResources.Italian + " )";
92:         //}
93:         //if (appSettings.AppLangName == "FR")
94:         //{
95:         //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.French + " )";
96:         //}
97:         //if (appSettings.AppLangName == "RU")
98:         //{
99:         //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Russian + " )";
100:        //}
101:        //if (appSettings.AppLangName == "ZH")
102:        //{
103:        //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Chinese + " )";
104:        //}
105:        //if (appSettings.AppLangName == "JA")
106:        //{
107:        //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Japanese + " )";
108:        //}
109:        //if (appSettings.AppLangName == "SA")
110:        //{
111:        //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Sanskrit + " )";
112:        //}
113:        //if (appSettings.AppLangName == "TH")
114:        //{
115:        //    lblLanguage.Text = AppResources.Language + " (" + AppResources.Selected
+ " : " + AppResources.Thai + " )";
116:        //}
117:
118:        if (appSettings.AppBackgroundColor == "BLA")
119:        {
120:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Black + " )";
121:        }
122:        if (appSettings.AppBackgroundColor == "BLU")
123:        {
124:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Blue + " )";
125:        }
126:        if (appSettings.AppBackgroundColor == "BRO")
127:        {
128:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Brown + " )";
129:        }
130:        if (appSettings.AppBackgroundColor == "RED")
131:        {
132:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Red + " )";
133:        }
134:        if (appSettings.AppBackgroundColor == "GRE")
135:        {
136:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Green + " )";
137:        }
138:        if (appSettings.AppBackgroundColor == "YEL")
139:        {
140:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Yellow + " )";
141:        }
142:        if (appSettings.AppBackgroundColor == "GRA")
143:        {
144:            lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + " : " + AppResources.Gray + " )";
145:        }
146:        if (appSettings.AppBackgroundColor == "ORA")

```



```

147:         {
148:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Orange + ")";
149:         }
150:         if (appSettings.AppBackgroundColor == "PUR")
151:         {
152:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Purple + ")";
153:         }
154:     }
155: }

```

1.1.2.16.2 GeneralSettingsPage Methods

1.1.2.16.2.1 GeneralSettingsPage.OnNavigatedFrom Method

C#

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class GeneralSettingsPage.

Body Source

```

1: protected override void OnNavigatedFrom(NavigationEventArgs e)
2: {
3:     base.OnNavigatedFrom(e);
4:     //while (NavigationService.CanGoBack)
5:     //NavigationService.RemoveBackEntry();
6:
7: }

```

1.1.2.16.2.2 GeneralSettingsPage.OnNavigatedTo Method

C#

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

Description

This is OnNavigatedTo, a member of class GeneralSettingsPage.

Body Source

```

1: protected override void OnNavigatedTo(NavigationEventArgs e)
2: {
3:     base.OnNavigatedTo(e);
4:     SetBackgroundColor();
5:     //while (NavigationService.CanGoBack)
6:     //NavigationService.RemoveBackEntry();
7:
8: }

```

1.1.2.16.2.3 GeneralSettingsPage.ReadFile Method

C#

```
public string ReadFile(string filePath);
```

Description

This is ReadFile, a member of class GeneralSettingsPage.

Body Source

```

1: public string ReadFile(string filePath)
2: {
3:     var ResrouceStream = Application.GetResourceStream(new Uri(filePath,

```

```

UriKind.Relative));
4:         if (ResrouceStream != null)
5:         {
6:             Stream myFileStream = ResrouceStream.Stream;
7:             if (myFileStream.CanRead)
8:             {
9:                 StreamReader myStreamReader = new StreamReader(myFileStream);
10:
11:                 return myStreamReader.ReadToEnd();
12:             }
13:         }
14:         return "";
15:     }

```

1.1.2.17 KelimeAnlamlari Class

Class Hierarchy

AwesomeDictionary.KelimeAnlamlari

C#

```

[Table]
public class KelimeAnlamlari;

```





File

KelimeAnlamlari.cs (see page 120)

Description

This is class AwesomeDictionary.KelimeAnlamlari.

KelimeAnlamlari Properties

	Name	Description
	KelimeAnlamlarild (see page 46)	This is KelimeAnlamlarild, a member of class KelimeAnlamlari.
	KelimeAnlamlariMeaning (see page 46)	This is KelimeAnlamlariMeaning, a member of class KelimeAnlamlari.
	KelimeAnlamlariName (see page 47)	This is KelimeAnlamlariName, a member of class KelimeAnlamlari.
	KelimeAnlamlariNameMeaning (see page 47)	This is KelimeAnlamlariNameMeaning, a member of class KelimeAnlamlari.

1.1.2.17.1 KelimeAnlamlari Properties

1.1.2.17.1.1 KelimeAnlamlari.KelimeAnlamlarild Property

C#

```

[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int KelimeAnlamlariId;

```

Description

This is KelimeAnlamlarild, a member of class KelimeAnlamlari.

1.1.2.17.1.2 KelimeAnlamlari.KelimeAnlamlariMeaning Property

C#

```

[Column]
public string KelimeAnlamlariMeaning;

```

Description

This is KelimeAnlamlariMeaning, a member of class KelimeAnlamlari.

1.1.2.17.1.3 KelimeAnlamlari.KelimeAnlamlariName Property**C#**

```
[Column]
public string KelimeAnlamlariName;
```

Description

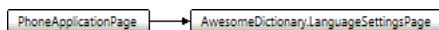
This is KelimeAnlamlariName, a member of class KelimeAnlamlari.

1.1.2.17.1.4 KelimeAnlamlari.KelimeAnlamlariNameMeaning Property**C#**

```
[Column]
public string KelimeAnlamlariNameMeaning;
```

Description

This is KelimeAnlamlariNameMeaning, a member of class KelimeAnlamlari.

1.1.2.18 LanguageSettingsPage Class**Class Hierarchy****C#**

```
public class LanguageSettingsPage : PhoneApplicationPage;
```

File

LanguageSettingsPage.xaml.cs (🔗 see page 121)

Description

This is class AwesomeDictionary.LanguageSettingsPage.

Methods

	Name	Description
🔗	LanguageSettingsPage (🔗 see page 47)	This is LanguageSettingsPage, a member of class LanguageSettingsPage.

LanguageSettingsPage Methods

	Name	Description
🔗	OnNavigatedFrom (🔗 see page 48)	This is OnNavigatedFrom, a member of class LanguageSettingsPage.
🔗	OnNavigatedTo (🔗 see page 48)	This is OnNavigatedTo, a member of class LanguageSettingsPage.

1.1.2.18.1 LanguageSettingsPage.LanguageSettingsPage Constructor**C#**

```
public LanguageSettingsPage();
```

Description

This is LanguageSettingsPage, a member of class LanguageSettingsPage.

Body Source

```
1: public LanguageSettingsPage()  
2: {  
3:     InitializeComponent();  
4:  
5:     lstLanguage.Items.Clear();  
6:     lstLanguage.Items.Add(AppResources.English);  
7:     lstLanguage.Items.Add(AppResources.Turkish);  
8:     lstLanguage.Items.Add(AppResources.German);  
9:     //lstLanguage.Items.Add(AppResources.Spanish);  
10:    //lstLanguage.Items.Add(AppResources.Russian);  
11:    //lstLanguage.Items.Add(AppResources.Arabic);  
12:    //lstLanguage.Items.Add(AppResources.Persian);  
13:    //lstLanguage.Items.Add(AppResources.Chinese);  
14:    //lstLanguage.Items.Add(AppResources.Italian);  
15:    //lstLanguage.Items.Add(AppResources.French);  
16:    //lstLanguage.Items.Add(AppResources.Japanese);  
17:    //lstLanguage.Items.Add(AppResources.Sanskrit);  
18:    //lstLanguage.Items.Add(AppResources.Thai);  
19:  
20:    lstLanguage.SelectedIndex = -1;  
21:    lblLanguage.Text = AppResources.SelectLanguage;  
22:    lblGeneralSettings.Text = AppResources.GeneralSettings;  
23:  
24:    SetBackgroundColor();  
25: }
```

1.1.2.18.2 LanguageSettingsPage Methods

1.1.2.18.2.1 LanguageSettingsPage.OnNavigatedFrom Method

C#

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class LanguageSettingsPage.

Body Source

```
1: protected override void OnNavigatedFrom(NavigationEventArgs e)  
2: {  
3:     base.OnNavigatedFrom(e);  
4: }
```

1.1.2.18.2.2 LanguageSettingsPage.OnNavigatedTo Method

C#

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

Description

This is OnNavigatedTo, a member of class LanguageSettingsPage.

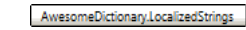
Body Source

```
1: protected override void OnNavigatedTo(NavigationEventArgs e)  
2: {  
3:     base.OnNavigatedTo(e);  
4:     SetBackgroundColor();  
5: }
```

1.1.2.19 LocalizedStrings Class

Provides access to string resources.

Class Hierarchy



C#

```
public class LocalizedStrings;
```

File

LocalizedStrings.cs (🔗 see page 125)

LocalizedStrings Properties

	Name	Description
	LocalizedResources (🔗 see page 49)	This is LocalizedResources, a member of class LocalizedStrings.

1.1.2.19.1 LocalizedStrings Properties

1.1.2.19.1.1 LocalizedStrings.LocalizedResources Property

C#

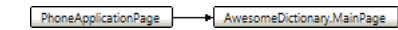
```
public AppResources LocalizedResources;
```

Description

This is LocalizedResources, a member of class LocalizedStrings.

1.1.2.20 MainPage Class

Class Hierarchy



C#

```
public class MainPage : PhoneApplicationPage;
```

File

MainPage.xaml.cs (🔗 see page 125)

Description

This is class AwesomeDictionary.MainPage.

Methods

	Name	Description
	MainPage (🔗 see page 49)	This is MainPage, a member of class MainPage.

1.1.2.20.1 MainPage.MainPage Constructor

C#

```
public MainPage();
```

Description

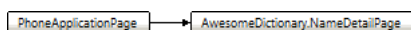
This is MainPage, a member of class MainPage.

Body Source

```

1: public MainPage()
2: {
3:     InitializeComponent();
4:
5:     ApplicationBar = new ApplicationBar();
6:
7:     ApplicationBarIconButton button2 = new ApplicationBarIconButton();
8:     button2.IconUri = new Uri("/Assets/Search.png", UriKind.Relative);
9:     button2.Text = AppResources.Search;
10:    ApplicationBar.Buttons.Add(button2);
11:    button2.Click += new EventHandler(SearchButton_Click);
12:
13:    ApplicationBarIconButton button3 = new ApplicationBarIconButton();
14:    button3.IconUri = new Uri("/Assets/Settings.png", UriKind.Relative);
15:    button3.Text = AppResources.Settings;
16:    ApplicationBar.Buttons.Add(button3);
17:    button3.Click += new EventHandler(SettingsButton_Click);
18:
19:    //ApplicationBarIconButton button4 = new ApplicationBarIconButton();
20:    //button4.IconUri = new Uri("/Assets/Statistics.png", UriKind.Relative);
21:    //button4.Text = AppResources.Statistics;
22:    //ApplicationBar.Buttons.Add(button4);
23:    //button4.Click += new EventHandler(StatisticsButton_Click);
24:
25:    ApplicationBarMenuItem menuItem1 = new ApplicationBarMenuItem();
26:    menuItem1.Text = AppResources.About;
27:    ApplicationBar.MenuItems.Add(menuItem1);
28:    menuItem1.Click += new EventHandler(AboutMenuItem_Click);
29:
30:    SetBackgroundColor();
31:
32:    piFavourite.Header = AppResources.MyFavourites;
33:    piRandomWords.Header = AppResources.RandomWords;
34:
35: }
```

1.1.2.21 NameDetailPage Class

Class Hierarchy**C#**

```
public class NameDetailPage : PhoneApplicationPage;
```

File

NameDetailPage.xaml.cs (see page 129)

Description

This is class AwesomeDictionary.NameDetailPage.

Methods

	Name	Description
	NameDetailPage (see page 51)	This is NameDetailPage, a member of class NameDetailPage.

NameDetailPage Fields

	Name	Description
🔗	flag (🔗 see page 52)	This is flag, a member of class NameDetailPage.
🔗	isFilled (🔗 see page 52)	This is isFilled, a member of class NameDetailPage.
🔗	pageName (🔗 see page 52)	This is pageName, a member of class NameDetailPage.
🔗	ratingValue (🔗 see page 52)	This is ratingValue, a member of class NameDetailPage.
🔗	wordId (🔗 see page 52)	This is wordId, a member of class NameDetailPage.

NameDetailPage Methods

	Name	Description
🔗	OnFragmentNavigation (🔗 see page 53)	This is OnFragmentNavigation, a member of class NameDetailPage.
🔗	OnNavigatedFrom (🔗 see page 54)	This is OnNavigatedFrom, a member of class NameDetailPage.
🔗	OnNavigatedTo (🔗 see page 54)	This is OnNavigatedTo, a member of class NameDetailPage.

1.1.2.21.1 NameDetailPage.NameDetailPage Constructor**C#**

```
public NameDetailPage();
```

Description

This is NameDetailPage, a member of class NameDetailPage.

Body Source

```
1: public NameDetailPage()
2: {
3:     InitializeComponent();
4:
5:     ApplicationBar = new ApplicationBar();
6:
7:     //ApplicationBarIconButton button1 = new ApplicationBarIconButton();
8:     //button1.IconUri = new Uri("/Assets/Save.png", UriKind.Relative);
9:     //button1.Text = "Kaydet";
10:    //ApplicationBar.Buttons.Add(button1);
11:    //button1.Click += new EventHandler(SaveButton_Click);
12:
13:    ApplicationBarIconButton button2 = new ApplicationBarIconButton();
14:    button2.IconUri = new Uri("/Assets/SendWithMail.png", UriKind.Relative);
15:    button2.Text = AppResources.SendWithEmail;
16:    ApplicationBar.Buttons.Add(button2);
17:    button2.Click += new EventHandler(SendMailButton_Click);
18:
19:    ApplicationBarIconButton button3 = new ApplicationBarIconButton();
20:    button3.IconUri = new Uri("/Assets/SendWithSMS.png", UriKind.Relative);
21:    button3.Text = AppResources.SendWithSMS;
22:    ApplicationBar.Buttons.Add(button3);
23:    button3.Click += new EventHandler(SendSMSButton_Click);
24:
25:    ApplicationBarIconButton button4 = new ApplicationBarIconButton();
26:    button4.IconUri = new Uri("/Assets/Share.png", UriKind.Relative);
27:    button4.Text = AppResources.Share;
28:    ApplicationBar.Buttons.Add(button4);
29:    button4.Click += new EventHandler(ShareNameButton_Click);
30:
31:    isFilled = false;
32:
33:    //ApplicationBarMenuItem menuItem1 = new ApplicationBarMenuItem();
34:    //menuItem1.Text = "Sil";
35:    //ApplicationBar.MenuItems.Add(menuItem1);
```

```
36:      //menuItem1.Click += new EventHandler(DeleteNameMenuItem_Click);
37:
38:      //List<string> genderList = new List<string>();
39:      //genderList.Add("Lütfen seçiniz");
40:      //genderList.Add("Erkek");
41:      //genderList.Add("Kadin");
42:      //genderList.Add("Erkek-Kadin");
43:      //lpGender.ItemsSource = genderList;
44:      //lpGender.SelectedIndex = 0;
45:
46:      SetBackgroundColor();
47: }
```

1.1.2.21.2 NameDetailPage Fields

1.1.2.21.2.1 NameDetailPage.flag Field

C#

```
public bool flag;
```

Description

This is flag, a member of class NameDetailPage.

1.1.2.21.2.2 NameDetailPage.isFilled Field

C#

```
public bool isFilled;
```

Description

This is isFilled, a member of class NameDetailPage.

1.1.2.21.2.3 NameDetailPage.pageName Field

C#

```
public string pageName;
```

Description

This is pageName, a member of class NameDetailPage.

1.1.2.21.2.4 NameDetailPage.ratingValue Field

C#

```
public double ratingValue = 0;
```

Description

This is ratingValue, a member of class NameDetailPage.

1.1.2.21.2.5 NameDetailPage.wordId Field

C#

```
public int wordId;
```

Description

This is wordId, a member of class NameDetailPage.

1.1.2.21.3 NameDetailPage Methods

1.1.2.21.3.1 NameDetailPage.OnFragmentNavigation Method

C#

```
protected override void OnFragmentNavigation(FragmentNavigationEventArgs e);
```

Description

This is OnFragmentNavigation, a member of class NameDetailPage.

Body Source

```
1: protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
2: {
3:     // displays "Fragment: Detail"
4:     //MessageBox.Show("Folder Id: " + e.Fragment);
5:     base.OnFragmentNavigation(e);
6:     //string fragmentName = e.Fragment.ToString();
7:     wordId = Convert.ToInt32(e.Fragment);
8:     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
9:     {
10:         var name = context.AllNames.Where(j =>
j.AllId.Equals(wordId)).SingleOrDefault() as All;
11:         allName = name;
12:         //for (int i = 0; i < lpGender.Items.Count; i++)
13:         //{
14:             // if (lpGender.Items[i].ToString() == name.AllGender)
15:             // {
16:                 // lpGender.SelectedIndex = i;
17:                 // break;
18:             // }
19:         //}
20:         lblNameDetail.Text = name.AllName;
21:         txtMeaning.Text = name.AllMeaning + Environment.NewLine + Environment.NewLine +
AppResources.Source + ":" + name.AllSource + "";
22:         txtMeaning.IsEnabled = false;
23:         //txtMeaning.Text = name.AllMeaning;
24:
25:         var favourite = context.Favourites.Where(j =>
j.FavouriteAllId.Equals(wordId)).SingleOrDefault() as Favourite;
26:         if (favourite != null)
27:         {
28:             ApplicationBarItem menuItem4 = new ApplicationBarItem();
29:             menuItem4.Text = AppResources.RemoveFromFavourites;
30:             ApplicationBar.MenuItems.Add(menuItem4);
31:             menuItem4.Click += new EventHandler(RemoveFavouritesMenuItem_Click);
32:         }
33:         else
34:         {
35:             ApplicationBarItem menuItem3 = new ApplicationBarItem();
36:             menuItem3.Text = AppResources.AddToFavourites;
37:             ApplicationBar.MenuItems.Add(menuItem3);
38:             menuItem3.Click += new EventHandler(AddFavouritesMenuItem_Click);
39:         }
40:
41:         //var myUpdate = context.MyUpdates.Where(j =>
j.MyUpdateName.Equals(fragmentName)).SingleOrDefault() as MyUpdate; ;
42:         //if (myUpdate != null)
43:         //{
44:             // ApplicationBarItem menuItem2 = new ApplicationBarItem();
45:             // menuItem2.Text = "Sisteme Eklenmesi İçin Gönder";
46:             // ApplicationBar.MenuItems.Add(menuItem2);
47:             // menuItem2.Click += new EventHandler(SaveAndSendMenuItem_Click);
48:         }
```

```

49:         //}
50:
51:
52:         isFilled = true;
53:         //pvName.SelectedIndex = 0;
54:     }
55: }

```

1.1.2.21.3.2 NameDetailPage.OnNavigatedFrom Method

C#

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class NameDetailPage.

Body Source

```

1: protected override void OnNavigatedFrom(NavigationEventArgs e)
2: {
3:     base.OnNavigatedFrom(e);
4:     //while (NavigationService.CanGoBack)
5:     //NavigationService.RemoveBackEntry();
6:
7: }

```

1.1.2.21.3.3 NameDetailPage.OnNavigatedTo Method

C#

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

Description

This is OnNavigatedTo, a member of class NameDetailPage.

Body Source

```

1: protected override void OnNavigatedTo(NavigationEventArgs e)
2: {
3:     base.OnNavigatedTo(e);
4:
5:     var lastPage = NavigationService.BackStack.FirstOrDefault();
6:     pageName = lastPage.Source.ToString();
7:     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
8:     {
9:         var appSettings = context.AppSettings.First();
10:
11:         FontFamily temp = new FontFamily(appSettings.FontFamily);
12:         double fontsize = double.Parse(appSettings.FontSize);
13:         txtMeaning.FontFamily = temp;
14:         txtMeaning.FontSize = fontsize;
15:     }
16:
17:     txtMeaning.IsEnabled = false;
18:     isFilled = false;
19:     //SetBackgroundColor();
20:     //while (NavigationService.CanGoBack)
21:     //NavigationService.RemoveBackEntry();
22:
23: }

```

1.1.2.22 OxfordEnglishEnglish Class

Class Hierarchy

```
AwesomeDictionary.OxfordEnglishEnglish
```

C#

```
[Table]
public class OxfordEnglishEnglish;
```





File

OxfordEnglishEnglish.cs (🔗 see page 139)

Description

This is class AwesomeDictionary.OxfordEnglishEnglish.

OxfordEnglishEnglish Properties

	Name	Description
	OxfordId (🔗 see page 55)	This is OxfordId, a member of class OxfordEnglishEnglish.
	OxfordMeaning (🔗 see page 55)	This is OxfordMeaning, a member of class OxfordEnglishEnglish.
	OxfordName (🔗 see page 55)	This is OxfordName, a member of class OxfordEnglishEnglish.
	OxfordNameMeaning (🔗 see page 55)	This is OxfordNameMeaning, a member of class OxfordEnglishEnglish.

1.1.2.22.1 OxfordEnglishEnglish Properties

1.1.2.22.1.1 OxfordEnglishEnglish.OxfordId Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int OxfordId;
```

Description

This is OxfordId, a member of class OxfordEnglishEnglish.

1.1.2.22.1.2 OxfordEnglishEnglish.OxfordMeaning Property

C#

```
[Column]
public string OxfordMeaning;
```

Description

This is OxfordMeaning, a member of class OxfordEnglishEnglish.

1.1.2.22.1.3 OxfordEnglishEnglish.OxfordName Property

C#

```
[Column]
public string OxfordName;
```

Description

This is OxfordName, a member of class OxfordEnglishEnglish.

1.1.2.22.1.4 OxfordEnglishEnglish.OxfordNameMeaning Property

C#

```
[Column]
public string OxfordNameMeaning;
```

Description

This is OxfordNameMeaning, a member of class OxfordEnglishEnglish.

1.1.2.23 RisaleNur Class

Class Hierarchy

AwesomeDictionary.RisaleNur

C#

```
[Table]
public class RisaleNur;
```





File

RisaleNur.cs ([see page 139](#))

Description

This is class AwesomeDictionary.RisaleNur.

RisaleNur Properties

	Name	Description
	RisaleNurId (see page 56)	This is RisaleNurId, a member of class RisaleNur.
	RisaleNurMeaning (see page 56)	This is RisaleNurMeaning, a member of class RisaleNur.
	RisaleNurName (see page 56)	This is RisaleNurName, a member of class RisaleNur.
	RisaleNurNameMeaning (see page 57)	This is RisaleNurNameMeaning, a member of class RisaleNur.

1.1.2.23.1 RisaleNur Properties

1.1.2.23.1.1 RisaleNur.RisaleNurId Property

C#

```
[Column(IsPrimaryKey = true, IsDbGenerated = true, DbType = "INT NOT NULL Identity",
CanBeNull = false)]
public int RisaleNurId;
```

Description

This is RisaleNurId, a member of class RisaleNur.

1.1.2.23.1.2 RisaleNur.RisaleNurMeaning Property

C#

```
[Column]
public string RisaleNurMeaning;
```

Description

This is RisaleNurMeaning, a member of class RisaleNur.

1.1.2.23.1.3 RisaleNur.RisaleNurName Property

C#

```
[Column]
public string RisaleNurName;
```

Description

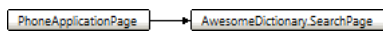
This is RisaleNurName, a member of class RisaleNur.

1.1.2.23.1.4 RisaleNur.RisaleNurNameMeaning Property**C#**

```
[Column]
public string RisaleNurNameMeaning;
```

Description

This is RisaleNurNameMeaning, a member of class RisaleNur.

1.1.2.24 SearchPage Class**Class Hierarchy****C#**

```
public class SearchPage : PhoneApplicationPage;
```

File

SearchPage.xaml.cs (🔗 see page 140)

Description

This is class AwesomeDictionary.SearchPage.

Methods

	Name	Description
🔗	SearchPage (🔗 see page 57)	This is SearchPage, a member of class SearchPage.

SearchPage Methods

	Name	Description
🔗	OnNavigatedFrom (🔗 see page 58)	This is OnNavigatedFrom, a member of class SearchPage.
🔗	OnNavigatedTo (🔗 see page 58)	This is OnNavigatedTo, a member of class SearchPage.

1.1.2.24.1 SearchPage.SearchPage Constructor**C#**

```
public SearchPage();
```

Description

This is SearchPage, a member of class SearchPage.

Body Source

```
1: public SearchPage()
2: {
3:     InitializeComponent();
4:     SetBackgroundColor();
5:
6:     txtSearchResult.Text = AppResources.SearchResults;
7:     txtSearchWithMeaning.Text = AppResources.SearchInMeanings;
8:     lblSearch.Text = AppResources.Search;
9:     //btnSearch.Content = AppResources.Search;
10:    //lstSearch.SelectedIndex = -1;
```

```
11: }
```

1.1.2.24.2 SearchPage Methods

1.1.2.24.2.1 SearchPage.OnNavigatedFrom Method

C#

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class SearchPage.

Body Source

```
1: protected override void OnNavigatedFrom(NavigationEventArgs e)
2: {
3:     base.OnNavigatedFrom(e);
4: }
```

1.1.2.24.2.2 SearchPage.OnNavigatedTo Method

C#

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

Description

This is OnNavigatedTo, a member of class SearchPage.

Body Source

```
1: protected override void OnNavigatedTo(NavigationEventArgs e)
2: {
3:     base.OnNavigatedTo(e);
4: }
```

1.1.2.25 StatisticsPage Class

Class Hierarchy



C#

```
public class StatisticsPage : PhoneApplicationPage;
```

File

StatisticsPage.xaml.cs (🔗 see page 144)

Description

This is class AwesomeDictionary.StatisticsPage.

Methods

	Name	Description
🔗	StatisticsPage (🔗 see page 59)	This is StatisticsPage, a member of class StatisticsPage.

StatisticsPage Methods

	Name	Description
🔗	OnNavigatedFrom (🔗 see page 59)	This is OnNavigatedFrom, a member of class StatisticsPage.
🔗	OnNavigatedTo (🔗 see page 59)	This is OnNavigatedTo, a member of class StatisticsPage.

1.1.2.25.1 StatisticsPage.StatisticsPage Constructor

C#

```
public StatisticsPage();
```

Description

This is StatisticsPage, a member of class StatisticsPage.

Body Source

```
1: public StatisticsPage()  
2: {  
3:     InitializeComponent();  
4:     lblStatistics.Text = AppResources.Statistics;  
5:     SetBackgroundColor();  
6: }
```

1.1.2.25.2 StatisticsPage Methods

1.1.2.25.2.1 StatisticsPage.OnNavigatedFrom Method

C#

```
protected override void OnNavigatedFrom(NavigationEventArgs e);
```

Description

This is OnNavigatedFrom, a member of class StatisticsPage.

Body Source

```
1: protected override void OnNavigatedFrom(NavigationEventArgs e)  
2: {  
3:     base.OnNavigatedFrom(e);  
4:     //while (NavigationService.CanGoBack)  
5:     //NavigationService.RemoveBackEntry();  
6:  
7: }
```

1.1.2.25.2.2 StatisticsPage.OnNavigatedTo Method

C#

```
protected override void OnNavigatedTo(NavigationEventArgs e);
```

Description

This is OnNavigatedTo, a member of class StatisticsPage.

Body Source

```
1: protected override void OnNavigatedTo(NavigationEventArgs e)  
2: {  
3:     base.OnNavigatedTo(e);  
4:     SetStatistic();  
5: }
```

1.2 Files

The following table lists files in this documentation.

Files

Name	Description
AboutPage.xaml.cs (see page 60)	This is file AboutPage.xaml.cs.
All.cs (see page 63)	This is file All.cs.
AlmancaTurkce.cs (see page 64)	This is file AlmancaTurkce.cs.
AlphaKeyGroup.cs (see page 64)	This is file AlphaKeyGroup.cs.
App.xaml.cs (see page 66)	This is file App.xaml.cs.
AppResources.Designer.cs (see page 73)	This code was generated by a tool. Runtime Version:4.0.30319.34014 Changes to this file may cause incorrect behavior and will be lost if the code is regenerated.
AppSettings.cs (see page 87)	This is file AppSettings.cs.
AssemblyInfo.cs (see page 87)	This is file AssemblyInfo.cs.
AwesomeDictionary.csproj (see page 88)	This is file AwesomeDictionary.csproj.
AwesomeDictionary.sln (see page 88)	This is file AwesomeDictionary.sln.
AwesomeDictionaryDataContext.cs (see page 88)	This is file AwesomeDictionaryDataContext.cs.
BackgroundColorSettingsPage.xaml.cs (see page 89)	This is file BackgroundColorSettingsPage.xaml.cs.
BilimSozlugu.cs (see page 92)	This is file BilimSozlugu.cs.
BuyukLugat.cs (see page 93)	This is file BuyukLugat.cs.
EnglishTurkishVol1.cs (see page 94)	This is file EnglishTurkishVol1.cs.
EnglishTurkishVol2.cs (see page 94)	This is file EnglishTurkishVol2.cs.
Favourite.cs (see page 95)	This is file Favourite.cs.
FontFamilySettingsPage.xaml.cs (see page 96)	This is file FontFamilySettingsPage.xaml.cs.
FontSizeSettingsPage.xaml.cs (see page 98)	This is file FontSizeSettingsPage.xaml.cs.
GeneralSettingsPage.xaml.cs (see page 101)	This is file GeneralSettingsPage.xaml.cs.
KelimeAnlamlari.cs (see page 120)	This is file KelimeAnlamlari.cs.
LanguageSettingsPage.xaml.cs (see page 121)	This is file LanguageSettingsPage.xaml.cs.
LocalizedStrings.cs (see page 125)	This is file LocalizedStrings.cs.
MainPage.xaml.cs (see page 125)	This is file MainPage.xaml.cs.
NameDetailPage.xaml.cs (see page 129)	This is file NameDetailPage.xaml.cs.
OxfordEnglishEnglish.cs (see page 139)	This is file OxfordEnglishEnglish.cs.
RisaleNur.cs (see page 139)	This is file RisaleNur.cs.
SearchPage.xaml.cs (see page 140)	This is file SearchPage.xaml.cs.
StatisticsPage.xaml.cs (see page 144)	This is file StatisticsPage.xaml.cs.

1.2.1 AboutPage.xaml.cs

This is file AboutPage.xaml.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Text;

```

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.


```

7: using System.Windows;
8: using System.Windows.Controls;
9: using System.Windows.Controls.Primitives;
10: using System.Windows.Input;
11: using System.Windows.Media;
12: using System.Windows.Media.Imaging;
13: using System.Windows.Navigation;
14: using Microsoft.Phone.Tasks;
15: using AwesomeDictionary.Resources;
16: using Microsoft.Phone.Controls;
17: using Microsoft.Phone.Shell;
18:
19: namespace AwesomeDictionary
20: {
21:     public partial class AboutPage : PhoneApplicationPage
22:     {
23:         public AboutPage()
24:         {
25:             InitializeComponent();
26:
27:             SetBackgroundColor();
28:
29:             ApplicationBar = new ApplicationBar();
30:
31:             ApplicationBarIconButton button2 = new ApplicationBarIconButton();
32:             button2.IconUri = new Uri("/Assets/SendWithMail.png", UriKind.Relative);
33:             button2.Text = AppResources.ContactWithUs;
34:             ApplicationBar.Buttons.Add(button2);
35:             button2.Click += new EventHandler(SendMailButton_Click);
36:
37:             ApplicationBarIconButton button3 = new ApplicationBarIconButton();
38:             button3.IconUri = new Uri("/Assets/Rate.png", UriKind.Relative);
39:             button3.Text = AppResources.Rate;
40:             ApplicationBar.Buttons.Add(button3);
41:             button3.Click += new EventHandler(RateButton_Click);
42:
43:             lblAboutTheApp.Text = AppResources.AboutTheApp;
44:             txtAbout.Text = AppResources.AboutTheAppText;
45:             txtAbout.IsEnabled = false;
46:             //var paragraph = new Paragraph();
47:             //paragraph.Inlines.Add(AppResources.AboutTheAppText);
48:             //txtAbout.Blocks.Add(paragraph);
49:         }
50:
51:         private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
52:         {
53:             if (this.NavigationService.CanGoBack)
54:             {
55:                 this.NavigationService.Navigate(new Uri("/MainPage.xaml",
UriKind.Relative));
56:             }
57:         }
58:
59:         private void SendMailButton_Click(object sender, EventArgs e)
60:         {
61:             // burada birden fazla e-posta hesabi varsa birini seçmesi söyleniyor
62:             //EmailAddressChooserTask emailAddressChooserTask;
63:             //emailAddressChooserTask = new EmailAddressChooserTask();
64:             //emailAddressChooserTask.Completed += new
EventHandler<EmailResult>(emailAddressChooserTask_Completed);
65:             //emailAddressChooserTask.Show();
66:             StringBuilder sb = new StringBuilder();
67:             EmailComposeTask emailComposeTask = new EmailComposeTask();
68:
69:
70:             sb.AppendLine();
71:             sb.AppendLine();
72:             sb.AppendLine(AppResources.SendWithAwesomeDictionaryApp);

```

```

73:
74:         emailComposeTask.Subject = AppResources.AboutAwesomeDictionary;
75:         emailComposeTask.Body = sb.ToString();
76:         emailComposeTask.To = "coderserdar@outlook.com";
77:         emailComposeTask.Cc = "";
78:         emailComposeTask.Bcc = "";
79:
80:         emailComposeTask.Show();
81:         //MessageBox.Show(AppResources.SuccessfulSendWithMail);
82:     }
83:
84:     private void RateButton_Click(object sender, EventArgs e)
85:     {
86:         MarketplaceReviewTask marketplaceReviewTask = new MarketplaceReviewTask();
87:         marketplaceReviewTask.Show();
88:     }
89:
90:     private void SetBackgroundColor()
91:     {
92:         AppSettings appSettings = new AppSettings();
93:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
94:         {
95:             appSettings = context.AppSettings.First() as AppSettings;
96:         }
97:
98:         if (appSettings.AppBackgroundImage != null)
99:         {
100:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
101:             BitmapImage image = new BitmapImage();
102:             image.SetSource(stream);
103:             ImageBrush ib = new ImageBrush();
104:             ib.ImageSource = image;
105:             this.LayoutRoot.Background = ib;
106:         }
107:         else
108:         {
109:             switch (appSettings.AppBackgroundColor)
110:             {
111:                 case "BLA":
112:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
113:                     break;
114:                 case "BLU":
115:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
116:                     break;
117:                 case "BRO":
118:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
119:                     break;
120:                 case "RED":
121:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
122:                     break;
123:                 case "GRE":
124:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
125:                     break;
126:                 case "GRA":
127:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
128:                     break;
129:                 case "YEL":
130:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
131:                     break;
132:                 case "ORA":
133:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
134:                     break;
135:                 case "PUR":
136:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
137:                     break;

```

```

138:                                     default:
139:                                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
140:                                     break;
141:                                 }
142:                             }
143:                         }
144:                     }
145: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.2 All.cs

This is file All.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8: using System.ComponentModel;
9: using Microsoft.Phone.Data.Linq;
10: // index özelliği aşağıdaki using sinifi ile etkili bir hale geliyor.
11: using Microsoft.Phone.Data.Linq.Mapping;
12:
13: namespace AwesomeDictionary
14: {
15:     [Index(Columns = "AllName, AllMeaning, AllNameMeaning, AllNameSource ASC", IsUnique
= false, Name = "indAllNames")]
16:     [Table]
17:     public class All
18:     {
19:         [Column(IsPrimaryKey = true,
20:             IsDbGenerated = true,
21:             DbType = "INT NOT NULL Identity",
22:             CanBeNull = false)]
23:         public int AllId { get; set; }
24:
25:         [Column]
26:         public string AllName { get; set; }
27:
28:         [Column]
29:         public string AllMeaning { get; set; }
30:
31:         [Column]
32:         public string AllNameMeaning { get; set; }
33:
34:         [Column]
35:         public string AllSource { get; set; }
36:
37:         [Column]
38:         public string AllNameSource { get; set; }
39:
40:     }
41: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.3 AlmancaTurkce.cs

This is file AlmancaTurkce.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class AlmancaTurkce
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int AlmancaTurkceId { get; set; }
19:
20:         [Column]
21:         public string AlmancaTurkceName { get; set; }
22:
23:         [Column]
24:         public string AlmancaTurkceMeaning { get; set; }
25:
26:         [Column]
27:         public string AlmancaTurkceNameMeaning { get; set; }
28:
29:     }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.4 AlphaKeyGroup.cs

This is file AlphaKeyGroup.cs.

Body Source

```
1: ?using System.Collections.Generic;
2: using System.Globalization;
3: using Microsoft.Phone.Globalization;
4:
5: namespace AwesomeDictionary
6: {
7:     public class AlphaKeyGroup<T> : List<T>
```

```

8:     {
9:         /// <summary>
10:        /// The delegate that is used to get the key information.
11:        /// </summary>
12:        /// <param name="item">An object of type T</param>
13:        /// <returns>The key value to use for this object</returns>
14:        public delegate string GetKeyDelegate(T item);
15:
16:        /// <summary>
17:        /// The Key of this group.
18:        /// </summary>
19:        public string Key { get; private set; }
20:
21:        /// <summary>
22:        /// Public constructor.
23:        /// </summary>
24:        /// <param name="key">The key for this group.</param>
25:        public AlphaKeyGroup(string key)
26:        {
27:            Key = key;
28:        }
29:
30:        /// <summary>
31:        /// Create a list of AlphaGroup<T> with keys set by a SortedLocaleGrouping.
32:        /// </summary>
33:        /// <param name="slg">The </param>
34:        /// <returns>The items source for a LongListSelector</returns>
35:        private static List<AlphaKeyGroup<T>> CreateGroups(SortedLocaleGrouping slg)
36:        {
37:            List<AlphaKeyGroup<T>> list = new List<AlphaKeyGroup<T>>();
38:
39:            foreach (string key in slg.GroupDisplayNames)
40:            {
41:                list.Add(new AlphaKeyGroup<T>(key));
42:            }
43:
44:            return list;
45:        }
46:
47:        /// <summary>
48:        /// Create a list of AlphaGroup<T> with keys set by a SortedLocaleGrouping.
49:        /// </summary>
50:        /// <param name="items">The items to place in the groups.</param>
51:        /// <param name="ci">The CultureInfo to group and sort by.</param>
52:        /// <param name="getKey">A delegate to get the key from an item.</param>
53:        /// <param name="sort">Will sort the data if true.</param>
54:        /// <returns>An items source for a LongListSelector</returns>
55:        public static List<AlphaKeyGroup<T>> CreateGroups(IEnumerable<T> items,
56:        CultureInfo ci, GetKeyDelegate getKey, bool sort)
57:        {
58:            SortedLocaleGrouping slg = new SortedLocaleGrouping(ci);
59:            List<AlphaKeyGroup<T>> list = CreateGroups(slg);
60:
61:            foreach (T item in items)
62:            {
63:                int index = 0;
64:                if (slg.SupportsPhonetics)
65:                {
66:                    //check if your database has yomi string for item
67:                    //if it does not, then do you want to generate Yomi or ask the user
68:                    //index = slg.GetGroupIndex(getKey(Yomiof(item)));
69:                }
70:                else
71:                {
72:                    index = slg.GetGroupIndex(getKey(item));
73:                }
74:                if (index >= 0 && index < list.Count)
75:                {

```

```
75:         list[index].Add(item);
76:     }
77: }
78:
79:     if (sort)
80:     {
81:         foreach (AlphaKeyGroup<T> group in list)
82:         {
83:             group.Sort((c0, c1) => { return ci.CompareInfo.Compare(getKey(c0),
84: getKey(c1)); });
85:         }
86:     }
87:     return list;
88: }
89:
90: }
91: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.5 App.xaml.cs

This is file App.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Diagnostics;
3: using System.Globalization;
4: using System.Linq;
5: using System.Resources;
6: using System.Threading;
7: using System.IO;
8: using System.Windows;
9: using System.Windows.Markup;
10: using System.Windows.Navigation;
11: using Microsoft.Phone.Controls;
12: using Microsoft.Phone.Shell;
13: using System.Collections.Generic;
14: using Microsoft.Phone.Marketplace;
15: using AwesomeDictionary.Resources;
16:
17: namespace AwesomeDictionary
18: {
19:     public partial class App : Application
20:     {
21:         /// <summary>
22:         /// Provides easy access to the root frame of the Phone Application.
23:         /// </summary>
24:         /// <returns>The root frame of the Phone Application.</returns>
25:         public static PhoneApplicationFrame RootFrame { get; private set; }
26:
27:         /// <summary>
28:         /// Constructor for the Application object.
29:         /// </summary>
30:         public App()
31:         {
32:             // Global handler for uncaught exceptions.
33:             UnhandledException += Application_UnhandledException;
34:
35:             // Standard XAML initialization
```

```

36:         InitializeComponent();
37:
38:         // ayarlardan temasi açık renk bile olsa
39:         // kapali gibi çalışmasını sağlayacak bir nuget paketi yüklendi
40:         // bu sorunu gideriyor
41:         ThemeManager.ToDarkTheme();
42:
43:         // Phone-specific initialization
44:         InitializePhoneApplication();
45:
46:         // Language display initialization
47:         InitializeLanguage();
48:
49:         // Show graphics profiling information while debugging.
50:         if (Debugger.IsAttached)
51:         {
52:             // Display the current frame rate counters.
53:             Application.Current.Host.Settings.EnableFrameRateCounter = true;
54:
55:             // Show the areas of the app that are being redrawn in each frame.
56:             //Application.Current.Host.Settings.EnableRedrawRegions = true;
57:
58:             // Enable non-production analysis visualization mode,
59:             // which shows areas of a page that are handed off to GPU with a
colored overlay.
60:             //Application.Current.Host.Settings.EnableCacheVisualization = true;
61:
62:             // Prevent the screen from turning off while under the debugger by
disabling
63:             // the application's idle detection.
64:             // Caution:- Use this under debug mode only. Application that disables
user idle detection will continue to run
65:             // and consume battery power when the user is not using the phone.
66:             PhoneApplicationService.Current.UserIdleDetectionMode =
IdleDetectionMode.Disabled;
67:         }
68:
69:     }
70:
71:     // Code to execute when the application is launching (eg, from Start)
72:     // This code will not execute when the application is reactivated
73:     private void Application_Launching(object sender, LaunchingEventArgs e)
74:     {
75:     }
76:
77:     // Code to execute when the application is activated (brought to foreground)
78:     // This code will not execute when the application is first launched
79:     private void Application_Activated(object sender, ActivatedEventArgs e)
80:     {
81:     }
82:
83:     // Code to execute when the application is deactivated (sent to background)
84:     // This code will not execute when the application is closing
85:     private void Application_Deactivated(object sender, DeactivatedEventArgs e)
86:     {
87:     }
88:
89:     // Code to execute when the application is closing (eg, user hit Back)
90:     // This code will not execute when the application is deactivated
91:     private void Application_Closing(object sender, ClosingEventArgs e)
92:     {
93:     }
94:
95:     // Code to execute if a navigation fails
96:     private void RootFrame_NavigationFailed(object sender,
NavigationFailedEventArgs e)
97:     {
98:         if (Debugger.IsAttached)
99:         {

```

```

100:         // A navigation has failed; break into the debugger
101:         Debugger.Break();
102:     }
103: }
104:
105: // Code to execute on Unhandled Exceptions
106: private void Application_UnhandledException(object sender,
ApplicationUnhandledExceptionEventArgs e)
107: {
108:     if (Debugger.IsAttached)
109:     {
110:         // An unhandled exception has occurred; break into the debugger
111:         Debugger.Break();
112:     }
113: }
114:
115: #region Phone application initialization
116:
117: // Avoid double-initialization
118: private bool phoneApplicationInitialized = false;
119:
120: // Do not add any additional code to this method
121: private void InitializePhoneApplication()
122: {
123:     if (phoneApplicationInitialized)
124:         return;
125:
126:     // Create the frame but don't set it as RootVisual yet; this allows the
splash
127:     // screen to remain active until the application is ready to render.
128:     RootFrame = new PhoneApplicationFrame();
129:     RootFrame.Navigated += CompleteInitializePhoneApplication;
130:
131:     // Handle navigation failures
132:     RootFrame.NavigationFailed += RootFrame_NavigationFailed;
133:
134:     // Handle reset requests for clearing the backstack
135:     RootFrame.Navigated += CheckForResetNavigation;
136:
137:     // Ensure we don't initialize again
138:     phoneApplicationInitialized = true;
139: }
140:
141: // Do not add any additional code to this method
142: private void CompleteInitializePhoneApplication(object sender,
NavigationEventArgs e)
143: {
144:     // Set the root visual to allow the application to render
145:     if (RootVisual != RootFrame)
146:         RootVisual = RootFrame;
147:
148:     // Remove this handler since it is no longer needed
149:     RootFrame.Navigated -= CompleteInitializePhoneApplication;
150: }
151:
152: private void CheckForResetNavigation(object sender, NavigationEventArgs e)
153: {
154:     // If the app has received a 'reset' navigation, then we need to check
155:     // on the next navigation to see if the page stack should be reset
156:     if (e.NavigationMode == NavigationMode.Reset)
157:         RootFrame.Navigated += ClearBackStackAfterReset;
158: }
159:
160: private void ClearBackStackAfterReset(object sender, NavigationEventArgs e)
161: {
162:     // Unregister the event so it doesn't get called again
163:     RootFrame.Navigated -= ClearBackStackAfterReset;
164:
165:     // Only clear the stack for 'new' (forward) and 'refresh' navigations

```



```

166:         if (e.NavigationMode != NavigationMode.New && e.NavigationMode !=
NavigationMode.Refresh)
167:             return;
168:
169:         // For UI consistency, clear the entire page stack
170:         while (RootFrame.RemoveBackEntry() != null)
171:         {
172:             ; // do nothing
173:         }
174:     }
175:
176:     #endregion
177:
178:     // Initialize the app's font and flow direction as defined in its localized
resource strings.
179:     //
180:     // To ensure that the font of your application is aligned with its supported
languages and that the
181:     // FlowDirection for each of those languages follows its traditional
direction, ResourceLanguage
182:     // and ResourceFlowDirection should be initialized in each resx file to match
these values with that
183:     // file's culture. For example:
184:     //
185:     // AppResources.es-ES.resx
186:     //     ResourceLanguage's value should be "es-ES"
187:     //     ResourceFlowDirection's value should be "LeftToRight"
188:     //
189:     // AppResources.ar-SA.resx
190:     //     ResourceLanguage's value should be "ar-SA"
191:     //     ResourceFlowDirection's value should be "RightToLeft"
192:     //
193:     // For more info on localizing Windows Phone apps see
http://go.microsoft.com/fwlink/?LinkId=262072.
194:     //
195:     private void InitializeLanguage()
196:     {
197:         try
198:         {
199:             // Set the font to match the display language defined by the
200:             // ResourceLanguage resource string for each supported language.
201:             //
202:             // Fall back to the font of the neutral language if the Display
203:             // language of the phone is not supported.
204:             //
205:             // If a compiler error is hit then ResourceLanguage is missing from
206:             // the resource file.
207:             RootFrame.Language =
XmlLanguage.GetLanguage(AppResources.ResourceLanguage);
208:
209:             // Set the FlowDirection of all elements under the root frame based
210:             // on the ResourceFlowDirection resource string for each
211:             // supported language.
212:             //
213:             // If a compiler error is hit then ResourceFlowDirection is missing
from
214:             // the resource file.
215:             FlowDirection flow = (FlowDirection)Enum.Parse(typeof(FlowDirection),
AppResources.ResourceFlowDirection);
216:             RootFrame.FlowDirection = flow;
217:         }
218:         catch
219:         {
220:             // If an exception is caught here it is most likely due to either
221:             // ResourceLanguage not being correctly set to a supported language
222:             // code or ResourceFlowDirection is set to a value other than
LeftToRight
223:             // or RightToLeft.
224:

```

```

225:         if (Debugger.IsAttached)
226:         {
227:             Debugger.Break();
228:         }
229:
230:         throw;
231:     }
232: }
233:
234: public string ReadFile(string filePath)
235: {
236:     var ResrouceStream = Application.GetResourceStream(new Uri(filePath,
UriKind.Relative));
237:     if (ResrouceStream != null)
238:     {
239:         Stream myFileStream = ResrouceStream.Stream;
240:         if (myFileStream.CanRead)
241:         {
242:             StreamReader myStreamReader = new StreamReader(myFileStream);
243:
244:             return myStreamReader.ReadToEnd();
245:         }
246:     }
247:     return "";
248: }
249:
250: private static void DilAyariOlustur(AwesomeDictionaryDataContext context)
251: {
252:     var appSettings = new AppSettings()
253:     {
254:         //AppLangId = 42,
255:         AppLangName = "EN",
256:         AppBackgroundColor = "BLA",
257:         FontFamily = "Verdana",
258:         FontSize = "30",
259:         AppBackgroundImage = null
260:     };
261:
262:     context.AppSettings.InsertOnSubmit(appSettings);
263:     context.SubmitChanges();
264:
265:     CultureInfo newCulture = new CultureInfo("en");
266:     Thread.CurrentThread.CurrentCulture = newCulture;
267:     Thread.CurrentThread.CurrentUICulture = newCulture;
268: }
269:
270: private void Application_Startup(object sender, StartupEventArgs e)
271: {
272:     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
273:     {
274:         if (!context.DatabaseExists())
275:         {
276:             context.CreateDatabase();
277:             DilAyariOlustur(context);
278:
279:             // burada metin belgeleri okunuyor ve veritabanındaki tablolara
ekleme yapiliyor
280:             //var male = ReadFile(@"Sources\ErkekAdlari.txt");
281:             //var maleLines = male.Split('\n').ToList() as List<string>;
282:             //for (int i = 1; i < maleLines.Count; i++)
283:             //{
284:             //    var information = maleLines[i].Split('_').ToList() as
List<string>;
285:             //    Male maleName = new Male();
286:             //    maleName.MaleName =
information[0].ToUpper().TrimEnd().TrimStart();
287:             //    maleName.MaleMeaning = information[1].TrimEnd().TrimStart();
288:

```

```

289:         // context.MaleNames.InsertOnSubmit(maleName);
290:     //}
291:
292:     //var female = ReadFile(@"Sources\KizAdlari.txt");
293:     //var femaleLines = female.Split('\n').ToList() as List<string>;
294:     //for (int i = 1; i < femaleLines.Count; i++)
295:     //{
296:         // var information = femaleLines[i].Split('_').ToList() as
List<string>;
297:         // Female femaleName = new Female();
298:         // femaleName.FemaleName =
information[0].ToUpper().TrimEnd().TrimStart();
299:         // femaleName.FemaleMeaning =
information[1].TrimEnd().TrimStart();
300:
301:         // context.FemaleNames.InsertOnSubmit(femaleName);
302:     //}
303:
304:     //context.SubmitChanges();
305:
306:     //var maleNamesTemp = context.MaleNames.ToList() as List<Male>;
307:     //var femaleNamesTemp = context.FemaleNames.ToList() as
List<Female>;
308:
309:     //for (int i = 0; i < maleNamesTemp.Count; i++)
310:     //{
311:         // for (int k = 0; k < femaleNamesTemp.Count; k++)
312:         // {
313:             // if (maleNamesTemp[i].MaleName ==
femaleNamesTemp[k].FemaleName)
314:             // {
315:                 // Unisex unisex = new Unisex();
316:                 // unisex.UnisexName = maleNamesTemp[i].MaleName;
317:                 // unisex.UnisexMeaning = maleNamesTemp[i].MaleMeaning;
318:                 // context.UnisexNames.InsertOnSubmit(unisex);
319:                 // var maleNameTemp = context.MaleNames.Where(j =>
j.MaleId.Equals(maleNamesTemp[i].MaleId)).SingleOrDefault() as Male;
320:                 // context.MaleNames.DeleteOnSubmit(maleNameTemp);
321:                 // var femaleNameTemp = context.FemaleNames.Where(j =>
j.FemaleId.Equals(femaleNamesTemp[k].FemaleId)).SingleOrDefault() as Female;
322:                 // context.FemaleNames.DeleteOnSubmit(femaleNameTemp);
323:                 // break;
324:             // }
325:         // }
326:     //}
327:
328:     //context.SubmitChanges();
329:
330:     //var males = context.MaleNames.ToList() as List<Male>;
331:     //for (int i = 0; i < males.Count; i++)
332:     //{
333:         // All allName = new All();
334:         // allName.AllName = males[i].MaleName;
335:         // allName.AllMeaning = males[i].MaleMeaning;
336:         // allName.AllNameMeaning = males[i].MaleName + " " +
males[i].MaleMeaning;
337:         // allName.AllGender = "Erkek";
338:
339:         // context.AllNames.InsertOnSubmit(allName);
340:     //}
341:
342:     //var females = context.FemaleNames.ToList() as List<Female>;
343:     //for (int i = 0; i < females.Count; i++)
344:     //{
345:         // All allName = new All();
346:         // allName.AllName = females[i].FemaleName;
347:         // allName.AllMeaning = females[i].FemaleMeaning;
348:         // allName.AllNameMeaning = females[i].FemaleName + " " +
females[i].FemaleMeaning;

```

```

349:         // allName.AllGender = "Kadin";
350:
351:         // context.AllNames.InsertOnSubmit(allName);
352:     //}
353:
354:     //var unisexes = context.UnisexNames.ToList() as List<Unisex>;
355:     //for (int i = 0; i < unisexes.Count; i++)
356:     //{
357:         // All allName = new All();
358:         // allName.AllName = unisexes[i].UnisexName;
359:         // allName.AllMeaning = unisexes[i].UnisexMeaning;
360:         // allName.AllNameMeaning = unisexes[i].UnisexName + " " +
unisexes[i].UnisexMeaning;
361:         // allName.AllGender = "Erkek-Kadin";
362:
363:         // context.AllNames.InsertOnSubmit(allName);
364:     //}
365:
366:     //context.SubmitChanges();
367: }
368: else
369: {
370:     using (var context2 = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
371:     {
372:
373:         AppSettings lang =
374:             context2.AppSettings.First() as AppSettings;
375:         string culture = "";
376:         switch (lang.AppLangName)
377:         {
378:             case "TR":
379:                 culture = "tr";
380:                 break;
381:             case "EN":
382:                 culture = "en";
383:                 break;
384:             case "DE":
385:                 culture = "de";
386:                 break;
387:             case "ES":
388:                 culture = "es";
389:                 break;
390:             case "FR":
391:                 culture = "fr";
392:                 break;
393:             case "IT":
394:                 culture = "it";
395:                 break;
396:             case "AR":
397:                 culture = "ar";
398:                 break;
399:             case "FA":
400:                 culture = "fa-IR";
401:                 break;
402:             case "ZH":
403:                 culture = "zh";
404:                 break;
405:             case "PT":
406:                 culture = "pt";
407:                 break;
408:             case "RU":
409:                 culture = "ru";
410:                 break;
411:             case "JA":
412:                 culture = "ja";
413:                 break;
414:             case "SA":
415:                 culture = "sa";

```

```
416:         break;
417:     case "TH":
418:         culture = "th";
419:         break;
420:     default:
421:         culture = "tr-TR";
422:         break;
423:     }
424:     CultureInfo newCulture = new CultureInfo(culture);
425:     Thread.CurrentThread.CurrentCulture = newCulture;
426:     Thread.CurrentThread.CurrentUICulture = newCulture;
427: }
428: }
429:
430: // kullanıcının programla ilgili bilgilendirici notları kendi dilinde
görebilmesi için burada ekliyoruz.
431: }
432: }
433: }
434: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.6 AppResources.Designer.cs

This code was generated by a tool. Runtime Version:4.0.30319.34014

Changes to this file may cause incorrect behavior and will be lost if the code is regenerated.

Body Source

```
1:  ?//-----
2:  // <auto-generated>
3:  //     This code was generated by a tool.
4:  //     Runtime Version:4.0.30319.34014
5:  //
6:  //     Changes to this file may cause incorrect behavior and will be lost if
7:  //     the code is regenerated.
8:  // </auto-generated>
9:  //-----
10:
11: namespace AwesomeDictionary.Resources {
12:     using System;
13:
14:
15:     /// <summary>
16:     ///     A strongly-typed resource class, for looking up localized strings, etc.
17:     /// </summary>
18:     /// This class was auto-generated by the StronglyTypedResourceBuilder
19:     /// class via a tool like ResGen or Visual Studio.
20:     /// To add or remove a member, edit your .ResX file then rerun ResGen
21:     /// with the /str option, or rebuild your VS project.
22:
23: [global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.StronglyTypedResourceBuilder",
24: "4.0.0.0")]
25:     [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
26:     [global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
27:     public class AppResources {
28:
29:         private static global::System.Resources.ResourceManager resourceMan;
```

```

29:         private static global::System.Globalization.CultureInfo resourceCulture;
30:
31:
[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance",
"CA1811:AvoidUncalledPrivateCode")]
32:         internal AppResources() {
33:         }
34:
35:         /// <summary>
36:         ///     Returns the cached ResourceManager instance used by this class.
37:         /// </summary>
38:
[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.Editor
BrowsableState.Advanced)]
39:         public static global::System.Resources.ResourceManager ResourceManager {
40:             get {
41:                 if (object.ReferenceEquals(resourceMan, null)) {
42:                     global::System.Resources.ResourceManager temp = new
global::System.Resources.ResourceManager("AwesomeDictionary.Resources.AppResources",
typeof(AppResources).Assembly);
43:                     resourceMan = temp;
44:                 }
45:                 return resourceMan;
46:             }
47:         }
48:
49:         /// <summary>
50:         ///     Overrides the current thread's CurrentUICulture property for all
51:         ///     resource lookups using this strongly typed resource class.
52:         /// </summary>
53:
[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.Editor
BrowsableState.Advanced)]
54:         public static global::System.Globalization.CultureInfo Culture {
55:             get {
56:                 return resourceCulture;
57:             }
58:             set {
59:                 resourceCulture = value;
60:             }
61:         }
62:
63:         /// <summary>
64:         ///     Looks up a localized string similar to About.
65:         /// </summary>
66:         public static string About {
67:             get {
68:                 return ResourceManager.GetString("About", resourceCulture);
69:             }
70:         }
71:
72:         /// <summary>
73:         ///     Looks up a localized string similar to About Awesome Dictionary.
74:         /// </summary>
75:         public static string AboutAwesomeDictionary {
76:             get {
77:                 return ResourceManager.GetString("AboutAwesomeDictionary",
resourceCulture);
78:             }
79:         }
80:
81:         /// <summary>
82:         ///     Looks up a localized string similar to About The App.
83:         /// </summary>
84:         public static string AboutTheApp {
85:             get {
86:                 return ResourceManager.GetString("AboutTheApp", resourceCulture);
87:             }
88:         }

```

```
89:
90:     /// <summary>
91:     /// Looks up a localized string similar to Hi everybody. I am with you with
a new app. I investigate a lot of dictionary applications in Windows Phone and i tried to
make a simple app which you will like it. If you rate the app and send your thoughts to
coderserdar@outlook.com, I will be so appreciated to you. With my best regards. CoderSerdar.
92:     /// </summary>
93:     public static string AboutTheAppText {
94:         get {
95:             return ResourceManager.GetString("AboutTheAppText", resourceCulture);
96:         }
97:     }
98:
99:     /// <summary>
100:    /// Looks up a localized string similar to Add To Favourites.
101:    /// </summary>
102:    public static string AddToFavourites {
103:        get {
104:            return ResourceManager.GetString("AddToFavourites", resourceCulture);
105:        }
106:    }
107:
108:    /// <summary>
109:    /// Looks up a localized string similar to You Should Select At Least One
Dictionary.
110:    /// </summary>
111:    public static string AtLeastOneDictionary {
112:        get {
113:            return ResourceManager.GetString("AtLeastOneDictionary",
resourceCulture);
114:        }
115:    }
116:
117:    /// <summary>
118:    /// Looks up a localized string similar to Background.
119:    /// </summary>
120:    public static string Background {
121:        get {
122:            return ResourceManager.GetString("Background", resourceCulture);
123:        }
124:    }
125:
126:    /// <summary>
127:    /// Looks up a localized string similar to Background Color.
128:    /// </summary>
129:    public static string BackgroundColor {
130:        get {
131:            return ResourceManager.GetString("BackgroundColor", resourceCulture);
132:        }
133:    }
134:
135:    /// <summary>
136:    /// Looks up a localized string similar to Background Color Has Been Changed
Successfully.
137:    /// </summary>
138:    public static string BackgroundColorChangedSuccessfully {
139:        get {
140:            return ResourceManager.GetString("BackgroundColorChangedSuccessfully",
resourceCulture);
141:        }
142:    }
143:
144:    /// <summary>
145:    /// Looks up a localized string similar to Background Image.
146:    /// </summary>
147:    public static string BackgroundImage {
148:        get {
149:            return ResourceManager.GetString("BackgroundImage", resourceCulture);
150:        }
151:    }
```

```

151:     }
152:
153:     /// <summary>
154:     /// Looks up a localized string similar to Background Image Has Been Changed
Successfully.
155:     /// </summary>
156:     public static string BackgroundImageChangedSuccessfully {
157:         get {
158:             return ResourceManager.GetString("BackgroundImageChangedSuccessfully",
resourceCulture);
159:         }
160:     }
161:
162:     /// <summary>
163:     /// Looks up a localized string similar to Background Image Has Been Removed
Successfully.
164:     /// </summary>
165:     public static string BackgroundImageRemovedSuccessfully {
166:         get {
167:             return ResourceManager.GetString("BackgroundImageRemovedSuccessfully",
resourceCulture);
168:         }
169:     }
170:
171:     /// <summary>
172:     /// Looks up a localized string similar to Background Settings Has Been
Reset Successfully.
173:     /// </summary>
174:     public static string BackgroundSettingsResetSuccessfully {
175:         get {
176:             return
ResourceManager.GetString("BackgroundSettingsResetSuccessfully", resourceCulture);
177:         }
178:     }
179:
180:     /// <summary>
181:     /// Looks up a localized string similar to Black.
182:     /// </summary>
183:     public static string Black {
184:         get {
185:             return ResourceManager.GetString("Black", resourceCulture);
186:         }
187:     }
188:
189:     /// <summary>
190:     /// Looks up a localized string similar to Blue.
191:     /// </summary>
192:     public static string Blue {
193:         get {
194:             return ResourceManager.GetString("Blue", resourceCulture);
195:         }
196:     }
197:
198:     /// <summary>
199:     /// Looks up a localized string similar to Brown.
200:     /// </summary>
201:     public static string Brown {
202:         get {
203:             return ResourceManager.GetString("Brown", resourceCulture);
204:         }
205:     }
206:
207:     /// <summary>
208:     /// Looks up a localized string similar to Buyuk Lugat (Turkish->Turkish).
209:     /// </summary>
210:     public static string BuyukLugat {
211:         get {
212:             return ResourceManager.GetString("BuyukLugat", resourceCulture);
213:         }

```



```
214:     }
215:
216:     /// <summary>
217:     /// Looks up a localized string similar to Cancel.
218:     /// </summary>
219:     public static string Cancel {
220:         get {
221:             return ResourceManager.GetString("Cancel", resourceCulture);
222:         }
223:     }
224:
225:     /// <summary>
226:     /// Looks up a localized string similar to Computer Dictionary
227:     /// (English-&gt;Turkish).
228:     /// </summary>
229:     public static string ComputerDictionary {
230:         get {
231:             return ResourceManager.GetString("ComputerDictionary",
232: resourceCulture);
233:         }
234:     }
235:
236:     /// <summary>
237:     /// Looks up a localized string similar to Contact With Us.
238:     /// </summary>
239:     public static string ContactWithUs {
240:         get {
241:             return ResourceManager.GetString("ContactWithUs", resourceCulture);
242:         }
243:     }
244:
245:     /// <summary>
246:     /// Looks up a localized string similar to Dictionaries Have Been Installed
247:     /// Successfully.
248:     /// </summary>
249:     public static string DictionariesInstalledSuccessfully {
250:         get {
251:             return ResourceManager.GetString("DictionariesInstalledSuccessfully",
252: resourceCulture);
253:         }
254:     }
255:
256:     /// <summary>
257:     /// Looks up a localized string similar to Dictionaries Have Been
258:     /// Uninstalled Successfully.
259:     /// </summary>
260:     public static string DictionariesUninstalledSuccessfully {
261:         get {
262:             return
263: ResourceManager.GetString("DictionariesUninstalledSuccessfully", resourceCulture);
264:         }
265:     }
266:
267:     /// <summary>
268:     /// Looks up a localized string similar to Dictionary Install.
269:     /// </summary>
270:     public static string DictionaryInstall {
271:         get {
272:             return ResourceManager.GetString("DictionaryInstall", resourceCulture);
273:         }
274:     }
275:
276:     /// <summary>
277:     /// Looks up a localized string similar to English.
278:     /// </summary>
279:     public static string English {
280:         get {
281:             return ResourceManager.GetString("English", resourceCulture);
282:         }
283:     }
284: }
```

```
277:     }
278:
279:     /// <summary>
280:     /// Looks up a localized string similar to English-&gt;Turkish Dictionary
Vol. 1.
281:     /// </summary>
282:     public static string EnglishTurkishVol1 {
283:         get {
284:             return ResourceManager.GetString("EnglishTurkishVol1",
resourceCulture);
285:         }
286:     }
287:
288:     /// <summary>
289:     /// Looks up a localized string similar to English-&gt;Turkish Dictionary
Vol. 2.
290:     /// </summary>
291:     public static string EnglishTurkishVol2 {
292:         get {
293:             return ResourceManager.GetString("EnglishTurkishVol2",
resourceCulture);
294:         }
295:     }
296:
297:     /// <summary>
298:     /// Looks up a localized string similar to Exit App.
299:     /// </summary>
300:     public static string ExitApp {
301:         get {
302:             return ResourceManager.GetString("ExitApp", resourceCulture);
303:         }
304:     }
305:
306:     /// <summary>
307:     /// Looks up a localized string similar to Are You Sure To Exit The
Application?.
308:     /// </summary>
309:     public static string ExitAppQuestion {
310:         get {
311:             return ResourceManager.GetString("ExitAppQuestion", resourceCulture);
312:         }
313:     }
314:
315:     /// <summary>
316:     /// Looks up a localized string similar to Favourite Word Count.
317:     /// </summary>
318:     public static string FavouriteNameCount {
319:         get {
320:             return ResourceManager.GetString("FavouriteNameCount",
resourceCulture);
321:         }
322:     }
323:
324:     /// <summary>
325:     /// Looks up a localized string similar to Font.
326:     /// </summary>
327:     public static string Font {
328:         get {
329:             return ResourceManager.GetString("Font", resourceCulture);
330:         }
331:     }
332:
333:     /// <summary>
334:     /// Looks up a localized string similar to Font Family.
335:     /// </summary>
336:     public static string FontFamily {
337:         get {
338:             return ResourceManager.GetString("FontFamily", resourceCulture);
339:         }
340:     }
341: }
```

```
340:         }
341:
342:         /// <summary>
343:         /// Looks up a localized string similar to Font Family Has Been Changed
344:         /// </summary>
345:         public static string FontFamilyChangedSuccessfully {
346:             get {
347:                 return ResourceManager.GetString("FontFamilyChangedSuccessfully",
348: resourceCulture);
349:             }
350:
351:         /// <summary>
352:         /// Looks up a localized string similar to Font Size.
353:         /// </summary>
354:         public static string FontSize {
355:             get {
356:                 return ResourceManager.GetString("FontSize", resourceCulture);
357:             }
358:
359:
360:         /// <summary>
361:         /// Looks up a localized string similar to Font Size Has Been Changed
362:         /// </summary>
363:         public static string FontSizeChangedSuccessfully {
364:             get {
365:                 return ResourceManager.GetString("FontSizeChangedSuccessfully",
366: resourceCulture);
367:             }
368:
369:         /// <summary>
370:         /// Looks up a localized string similar to General Settings.
371:         /// </summary>
372:         public static string GeneralSettings {
373:             get {
374:                 return ResourceManager.GetString("GeneralSettings", resourceCulture);
375:             }
376:
377:
378:         /// <summary>
379:         /// Looks up a localized string similar to German.
380:         /// </summary>
381:         public static string German {
382:             get {
383:                 return ResourceManager.GetString("German", resourceCulture);
384:             }
385:
386:
387:         /// <summary>
388:         /// Looks up a localized string similar to German->Turkish Dictionary.
389:         /// </summary>
390:         public static string GermanTurkish {
391:             get {
392:                 return ResourceManager.GetString("GermanTurkish", resourceCulture);
393:             }
394:
395:
396:         /// <summary>
397:         /// Looks up a localized string similar to Gray.
398:         /// </summary>
399:         public static string Gray {
400:             get {
401:                 return ResourceManager.GetString("Gray", resourceCulture);
402:             }
403:
404:         }
```

```
405:         /// <summary>
406:         ///     Looks up a localized string similar to Green.
407:         /// </summary>
408:         public static string Green {
409:             get {
410:                 return ResourceManager.GetString("Green", resourceCulture);
411:             }
412:         }
413:
414:         /// <summary>
415:         ///     Looks up a localized string similar to Install.
416:         /// </summary>
417:         public static string Install {
418:             get {
419:                 return ResourceManager.GetString("Install", resourceCulture);
420:             }
421:         }
422:
423:         /// <summary>
424:         ///     Looks up a localized string similar to Installing.
425:         /// </summary>
426:         public static string Installing {
427:             get {
428:                 return ResourceManager.GetString("Installing", resourceCulture);
429:             }
430:         }
431:
432:         /// <summary>
433:         ///     Looks up a localized string similar to Language.
434:         /// </summary>
435:         public static string Language {
436:             get {
437:                 return ResourceManager.GetString("Language", resourceCulture);
438:             }
439:         }
440:
441:         /// <summary>
442:         ///     Looks up a localized string similar to You may restart the application
443:         for changes will be effect..
444:         /// </summary>
445:         public static string LanguageWarning {
446:             get {
447:                 return ResourceManager.GetString("LanguageWarning", resourceCulture);
448:             }
449:         }
450:
451:         /// <summary>
452:         ///     Looks up a localized string similar to Meaning.
453:         /// </summary>
454:         public static string Meaning {
455:             get {
456:                 return ResourceManager.GetString("Meaning", resourceCulture);
457:             }
458:         }
459:
460:         /// <summary>
461:         ///     Looks up a localized string similar to My Favourites.
462:         /// </summary>
463:         public static string MyFavourites {
464:             get {
465:                 return ResourceManager.GetString("MyFavourites", resourceCulture);
466:             }
467:         }
468:
469:         /// <summary>
470:         ///     Looks up a localized string similar to Ok.
471:         /// </summary>
472:         public static string Ok {
473:             get {
```

```
473:         return ResourceManager.GetString("Ok", resourceCulture);
474:     }
475: }
476:
477: /// <summary>
478: /// Looks up a localized string similar to Orange.
479: /// </summary>
480: public static string Orange {
481:     get {
482:         return ResourceManager.GetString("Orange", resourceCulture);
483:     }
484: }
485:
486: /// <summary>
487: /// Looks up a localized string similar to Oxford (English->English).
488: /// </summary>
489: public static string OxfordDictionary {
490:     get {
491:         return ResourceManager.GetString("OxfordDictionary", resourceCulture);
492:     }
493: }
494:
495: /// <summary>
496: /// Looks up a localized string similar to Purple.
497: /// </summary>
498: public static string Purple {
499:     get {
500:         return ResourceManager.GetString("Purple", resourceCulture);
501:     }
502: }
503:
504: /// <summary>
505: /// Looks up a localized string similar to Random 10 Words.
506: /// </summary>
507: public static string RandomWords {
508:     get {
509:         return ResourceManager.GetString("RandomWords", resourceCulture);
510:     }
511: }
512:
513: /// <summary>
514: /// Looks up a localized string similar to Rate.
515: /// </summary>
516: public static string Rate {
517:     get {
518:         return ResourceManager.GetString("Rate", resourceCulture);
519:     }
520: }
521:
522: /// <summary>
523: /// Looks up a localized string similar to Red.
524: /// </summary>
525: public static string Red {
526:     get {
527:         return ResourceManager.GetString("Red", resourceCulture);
528:     }
529: }
530:
531: /// <summary>
532: /// Looks up a localized string similar to Remove Background Image.
533: /// </summary>
534: public static string RemoveBackgroundImage {
535:     get {
536:         return ResourceManager.GetString("RemoveBackgroundImage",
resourceCulture);
537:     }
538: }
539:
540: /// <summary>
```

```
541:         /// Looks up a localized string similar to Remove From Favourites.
542:         /// </summary>
543:         public static string RemoveFromFavourite {
544:             get {
545:                 return ResourceManager.GetString("RemoveFromFavourite",
resourceCulture);
546:             }
547:         }
548:
549:         /// <summary>
550:         /// Looks up a localized string similar to You Will Remove The Word From
Favourites. Are You Sure?.
551:         /// </summary>
552:         public static string RemoveFromFavouriteQuestion {
553:             get {
554:                 return ResourceManager.GetString("RemoveFromFavouriteQuestion",
resourceCulture);
555:             }
556:         }
557:
558:         /// <summary>
559:         /// Looks up a localized string similar to Remove From Favourites.
560:         /// </summary>
561:         public static string RemoveFromFavourites {
562:             get {
563:                 return ResourceManager.GetString("RemoveFromFavourites",
resourceCulture);
564:             }
565:         }
566:
567:         /// <summary>
568:         /// Looks up a localized string similar to Reset Settings.
569:         /// </summary>
570:         public static string ResetSettings {
571:             get {
572:                 return ResourceManager.GetString("ResetSettings", resourceCulture);
573:             }
574:         }
575:
576:         /// <summary>
577:         /// Looks up a localized string similar to LeftToRight.
578:         /// </summary>
579:         public static string ResourceFlowDirection {
580:             get {
581:                 return ResourceManager.GetString("ResourceFlowDirection",
resourceCulture);
582:             }
583:         }
584:
585:         /// <summary>
586:         /// Looks up a localized string similar to en-US.
587:         /// </summary>
588:         public static string ResourceLanguage {
589:             get {
590:                 return ResourceManager.GetString("ResourceLanguage", resourceCulture);
591:             }
592:         }
593:
594:         /// <summary>
595:         /// Looks up a localized string similar to Risale Nur (Turkish->Turkish).
596:         /// </summary>
597:         public static string RisaleNur {
598:             get {
599:                 return ResourceManager.GetString("RisaleNur", resourceCulture);
600:             }
601:         }
602:
603:         /// <summary>
604:         /// Looks up a localized string similar to Search.
```

```
605:         /// </summary>
606:         public static string Search {
607:             get {
608:                 return ResourceManager.GetString("Search", resourceCulture);
609:             }
610:         }
611:
612:         /// <summary>
613:         /// Looks up a localized string similar to Search Completed.
614:         /// </summary>
615:         public static string SearchCompleted {
616:             get {
617:                 return ResourceManager.GetString("SearchCompleted", resourceCulture);
618:             }
619:         }
620:
621:         /// <summary>
622:         /// Looks up a localized string similar to Search In Meanings.
623:         /// </summary>
624:         public static string SearchInMeanings {
625:             get {
626:                 return ResourceManager.GetString("SearchInMeanings", resourceCulture);
627:             }
628:         }
629:
630:         /// <summary>
631:         /// Looks up a localized string similar to Search Results.
632:         /// </summary>
633:         public static string SearchResults {
634:             get {
635:                 return ResourceManager.GetString("SearchResults", resourceCulture);
636:             }
637:         }
638:
639:         /// <summary>
640:         /// Looks up a localized string similar to Search Criteria Can Not Be Empty.
641:         /// </summary>
642:         public static string SearchTrimFault {
643:             get {
644:                 return ResourceManager.GetString("SearchTrimFault", resourceCulture);
645:             }
646:         }
647:
648:         /// <summary>
649:         /// Looks up a localized string similar to Select.
650:         /// </summary>
651:         public static string Select {
652:             get {
653:                 return ResourceManager.GetString("Select", resourceCulture);
654:             }
655:         }
656:
657:         /// <summary>
658:         /// Looks up a localized string similar to Select Background Color.
659:         /// </summary>
660:         public static string SelectBackgroundColor {
661:             get {
662:                 return ResourceManager.GetString("SelectBackgroundColor",
resourceCulture);
663:             }
664:         }
665:
666:         /// <summary>
667:         /// Looks up a localized string similar to Selected.
668:         /// </summary>
669:         public static string Selected {
670:             get {
671:                 return ResourceManager.GetString("Selected", resourceCulture);
672:             }
673:         }
```

```
673:     }
674:
675:     /// <summary>
676:     /// Looks up a localized string similar to Select Font Family.
677:     /// </summary>
678:     public static string SelectFontFamily {
679:         get {
680:             return ResourceManager.GetString("SelectFontFamily", resourceCulture);
681:         }
682:     }
683:
684:     /// <summary>
685:     /// Looks up a localized string similar to Select Font Size.
686:     /// </summary>
687:     public static string SelectFontSize {
688:         get {
689:             return ResourceManager.GetString("SelectFontSize", resourceCulture);
690:         }
691:     }
692:
693:     /// <summary>
694:     /// Looks up a localized string similar to Select Language.
695:     /// </summary>
696:     public static string SelectLanguage {
697:         get {
698:             return ResourceManager.GetString("SelectLanguage", resourceCulture);
699:         }
700:     }
701:
702:     /// <summary>
703:     /// Looks up a localized string similar to Send With Awesome Dictionary App.
704:     /// </summary>
705:     public static string SendWithAwesomeDictionaryApp {
706:         get {
707:             return ResourceManager.GetString("SendWithAwesomeDictionaryApp",
resourceCulture);
708:         }
709:     }
710:
711:     /// <summary>
712:     /// Looks up a localized string similar to Send With E-Mail.
713:     /// </summary>
714:     public static string SendWithEmail {
715:         get {
716:             return ResourceManager.GetString("SendWithEmail", resourceCulture);
717:         }
718:     }
719:
720:     /// <summary>
721:     /// Looks up a localized string similar to Send With SMS.
722:     /// </summary>
723:     public static string SendWithSMS {
724:         get {
725:             return ResourceManager.GetString("SendWithSMS", resourceCulture);
726:         }
727:     }
728:
729:     /// <summary>
730:     /// Looks up a localized string similar to Settings.
731:     /// </summary>
732:     public static string Settings {
733:         get {
734:             return ResourceManager.GetString("Settings", resourceCulture);
735:         }
736:     }
737:
738:     /// <summary>
739:     /// Looks up a localized string similar to Share.
740:     /// </summary>
```



```
741:         public static string Share {
742:             get {
743:                 return ResourceManager.GetString("Share", resourceCulture);
744:             }
745:         }
746:
747:         /// <summary>
748:         /// Looks up a localized string similar to Source.
749:         /// </summary>
750:         public static string Source {
751:             get {
752:                 return ResourceManager.GetString("Source", resourceCulture);
753:             }
754:         }
755:
756:         /// <summary>
757:         /// Looks up a localized string similar to Statistics.
758:         /// </summary>
759:         public static string Statistics {
760:             get {
761:                 return ResourceManager.GetString("Statistics", resourceCulture);
762:             }
763:         }
764:
765:         /// <summary>
766:         /// Looks up a localized string similar to Synchronizing.
767:         /// </summary>
768:         public static string Synchronizing {
769:             get {
770:                 return ResourceManager.GetString("Synchronizing", resourceCulture);
771:             }
772:         }
773:
774:         /// <summary>
775:         /// Looks up a localized string similar to System Has A Fault. Please Try
Again Later..
776:         /// </summary>
777:         public static string SystemFault {
778:             get {
779:                 return ResourceManager.GetString("SystemFault", resourceCulture);
780:             }
781:         }
782:
783:         /// <summary>
784:         /// Looks up a localized string similar to Total Word Count.
785:         /// </summary>
786:         public static string TotalNameCount {
787:             get {
788:                 return ResourceManager.GetString("TotalNameCount", resourceCulture);
789:             }
790:         }
791:
792:         /// <summary>
793:         /// Looks up a localized string similar to Turkish.
794:         /// </summary>
795:         public static string Turkish {
796:             get {
797:                 return ResourceManager.GetString("Turkish", resourceCulture);
798:             }
799:         }
800:
801:         /// <summary>
802:         /// Looks up a localized string similar to Uninstall.
803:         /// </summary>
804:         public static string Uninstall {
805:             get {
806:                 return ResourceManager.GetString("Uninstall", resourceCulture);
807:             }
808:         }
```

```
809:
810:     /// <summary>
811:     /// Looks up a localized string similar to Word.
812:     /// </summary>
813:     public static string Word {
814:         get {
815:             return ResourceManager.GetString("Word", resourceCulture);
816:         }
817:     }
818:
819:     /// <summary>
820:     /// Looks up a localized string similar to Word Has Been Added To Favourites
821:     Successfully.
822:     /// </summary>
823:     public static string WordAddedFavouriteSuccessfully {
824:         get {
825:             return ResourceManager.GetString("WordAddedFavouriteSuccessfully",
826: resourceCulture);
827:         }
828:     }
829:     /// <summary>
830:     /// Looks up a localized string similar to Word Is Already Favourite.
831:     /// </summary>
832:     public static string WordAlreadyFavourite {
833:         get {
834:             return ResourceManager.GetString("WordAlreadyFavourite",
835: resourceCulture);
836:         }
837:     }
838:     /// <summary>
839:     /// Looks up a localized string similar to Word And Meaning.
840:     /// </summary>
841:     public static string WordAndMeaning {
842:         get {
843:             return ResourceManager.GetString("WordAndMeaning", resourceCulture);
844:         }
845:     }
846:     /// <summary>
847:     /// Looks up a localized string similar to Word Meaning Dictionary.
848:     /// </summary>
849:     public static string WordMeaning {
850:         get {
851:             return ResourceManager.GetString("WordMeaning", resourceCulture);
852:         }
853:     }
854:
855:     /// <summary>
856:     /// Looks up a localized string similar to Word Has Been Removed From
857:     Favourites Successfully.
858:     /// </summary>
859:     public static string WordRemovedFavouriteSuccessfully {
860:         get {
861:             return ResourceManager.GetString("WordRemovedFavouriteSuccessfully",
862: resourceCulture);
863:         }
864:     }
865:     /// <summary>
866:     /// Looks up a localized string similar to Yellow.
867:     /// </summary>
868:     public static string Yellow {
869:         get {
870:             return ResourceManager.GetString("Yellow", resourceCulture);
871:         }
872:     }
```

```
873: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.7 AppSettings.cs

This is file AppSettings.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class AppSettings
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int AppSettingsId { get; set; }
19:
20:         [Column]
21:         public string AppLangName { get; set; }
22:
23:         [Column]
24:         public string AppBackgroundColor { get; set; }
25:
26:         [Column]
27:         public string FontFamily { get; set; }
28:
29:         [Column]
30:         public string FontSize { get; set; }
31:
32:         [Column(DbType = "Image", UpdateCheck = UpdateCheck.Never)]
33:         public byte[] AppBackgroundImage { get; set; }
34:
35:         //[Column]
36:         //public string ActiveDictionary { get; set; }
37:     }
38: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.8 AssemblyInfo.cs

This is file AssemblyInfo.cs.

Body Source

```
1: ?using System.Reflection;
2: using System.Runtime.CompilerServices;
3: using System.Runtime.InteropServices;
4: using System.Resources;
5:
6: // General Information about an assembly is controlled through the following
7: // set of attributes. Change these attribute values to modify the information
8: // associated with an assembly.
9: [assembly: AssemblyTitle("Awesome Dictionary")]
10: [assembly: AssemblyDescription("Best Dictionary App Ever")]
11: [assembly: AssemblyConfiguration("")]
12: [assembly: AssemblyCompany("CoderSerdar")]
13: [assembly: AssemblyProduct("CoderSerdar")]
14: [assembly: AssemblyCopyright("Copyright © 2015")]
15: [assembly: AssemblyTrademark("")]
16: [assembly: AssemblyCulture("")]
17:
18: // Setting ComVisible to false makes the types in this assembly not visible
19: // to COM components. If you need to access a type in this assembly from
20: // COM, set the ComVisible attribute to true on that type.
21: [assembly: ComVisible(false)]
22:
23: // The following GUID is for the ID of the typelib if this project is exposed to COM
24: [assembly: Guid("6e32f667-ec90-4f15-b44d-4bd7611c7ec4")]
25:
26: // Version information for an assembly consists of the following four values:
27: //
28: //      Major Version
29: //      Minor Version
30: //      Build Number
31: //      Revision
32: //
33: // You can specify all the values or you can default the Revision and Build Numbers
34: // by using the '*' as shown below:
35: [assembly: AssemblyVersion("1.0.0.1")]
36: [assembly: AssemblyFileVersion("1.0.0.1")]
37: [assembly: NeutralResourcesLanguageAttribute("en-US")]
```

1.2.9 AwesomeDictionary.csproj

This is file AwesomeDictionary.csproj.

1.2.10 AwesomeDictionary.sln

This is file AwesomeDictionary.sln.

1.2.11 AwesomeDictionaryDataContext.cs

This is file AwesomeDictionaryDataContext.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
```

```

4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     public class AwesomeDictionaryDataContext : DataContext
12:     {
13:         public const string ConnectionString = @"Data
Source=isostore:/AwesomeDictionary.sdf; Max Database Size=256; Max Buffer Size=4096;";
14:         public AwesomeDictionaryDataContext(string connectionString)
15:             : base(connectionString) { }
16:         public Table<AlmancaTurkce> AlmancaTurkces;
17:         public Table<RisaleNur> RisaleNurs;
18:         public Table<BilisimSozlugu> Bilisims;
19:         public Table<All> AllNames;
20:         public Table<Favourite> Favourites;
21:         public Table<EnglishTurkishVol1> EnglishTurkishVolls;
22:         public Table<EnglishTurkishVol2> EnglishTurkishVol2s;
23:         public Table<BuyukLugat> BuyukLugats;
24:         public Table<OxfordEnglishEnglish> Oxfords;
25:         public Table<KelimeAnlamlari> Kelimes;
26:         public Table<AppSettings> AppSettings;
27:     }
28: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.12 BackgroundColorSettingsPage.xaml.cs

This is file BackgroundColorSettingsPage.xaml.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Windows;
7: using System.Windows.Controls;
8: using System.Windows.Media;
9: using System.Windows.Media.Imaging;
10: using System.Windows.Navigation;
11: using AwesomeDictionary.Resources;
12: using Microsoft.Phone.Controls;
13: using Microsoft.Phone.Shell;
14:
15: namespace AwesomeDictionary
16: {
17:     public partial class BackgroundColorSettingsPage : PhoneApplicationPage
18:     {
19:         public BackgroundColorSettingsPage()
20:         {
21:             InitializeComponent();
22:
23:             lstBackgroundColor.Items.Clear();
24:             lstBackgroundColor.Items.Add(AppResources.Black);
25:             lstBackgroundColor.Items.Add(AppResources.Blue);
26:             lstBackgroundColor.Items.Add(AppResources.Brown);
27:             lstBackgroundColor.Items.Add(AppResources.Gray);

```

```

28:         lstBackgroundColor.Items.Add(AppResources.Green);
29:         lstBackgroundColor.Items.Add(AppResources.Orange);
30:         lstBackgroundColor.Items.Add(AppResources.Purple);
31:         lstBackgroundColor.Items.Add(AppResources.Red);
32:         lstBackgroundColor.Items.Add(AppResources.Yellow);
33:         lstBackgroundColor.SelectedIndex = -1;
34:
35:         lblBackgroundColor.Text = AppResources.SelectBackgroundColor;
36:         lblGeneralSettings.Text = AppResources.GeneralSettings;
37:
38:         SetBackgroundColor();
39:     }
40:
41:     private void SetBackgroundColor()
42:     {
43:         AppSettings appSettings = new AppSettings();
44:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
45:         {
46:             appSettings = context.AppSettings.First() as AppSettings;
47:         }
48:
49:         if (appSettings.AppBackgroundImage != null)
50:         {
51:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
52:             BitmapImage image = new BitmapImage();
53:             image.SetSource(stream);
54:             ImageBrush ib = new ImageBrush();
55:             ib.ImageSource = image;
56:             this.LayoutRoot.Background = ib;
57:         }
58:         else
59:         {
60:             switch (appSettings.AppBackgroundColor)
61:             {
62:                 case "BLA":
63:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
64:                     break;
65:                 case "BLU":
66:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
67:                     break;
68:                 case "BRO":
69:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
70:                     break;
71:                 case "RED":
72:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
73:                     break;
74:                 case "GRE":
75:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
76:                     break;
77:                 case "GRA":
78:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
79:                     break;
80:                 case "YEL":
81:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
82:                     break;
83:                 case "ORA":
84:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
85:                     break;
86:                 case "PUR":
87:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
88:                     break;
89:                 default:
90:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
91:                     break;
92:             }

```

```
93:         }
94:     }
95:
96:     protected override void OnNavigatedTo(NavigationEventArgs e)
97:     {
98:         base.OnNavigatedTo(e);
99:         //SetBackgroundColor();
100:        //while (NavigationService.CanGoBack)
101:        //NavigationService.RemoveBackEntry();
102:    }
103:
104:
105:     protected override void OnNavigatedFrom(NavigationEventArgs e)
106:     {
107:         base.OnNavigatedFrom(e);
108:         //while (NavigationService.CanGoBack)
109:         //NavigationService.RemoveBackEntry();
110:    }
111:
112:
113:     protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
114:     {
115:         // displays "Fragment: Detail"
116:         //MessageBox.Show("Folder Id: " + e.Fragment);
117:         // base.OnFragmentNavigation(e);
118:         // using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
119:         // {
120:         //     lblGeneralSettings.Text = AppResources.GeneralSettings;
121:         //     lblBackgroundColor.Text = AppResources.SelectFontSize;
122:         // }
123:    }
124:
125:     private void lstBackgroundColor_SelectionChanged(object sender,
SelectionChangedEventArgs e)
126:     {
127:         int index = lstBackgroundColor.SelectedIndex;
128:         string backgroundColor = "";
129:         if (index == 0)
130:         {
131:             backgroundColor = "BLA";
132:         }
133:         else if (index == 1)
134:         {
135:             backgroundColor = "BLU";
136:         }
137:         else if (index == 2)
138:         {
139:             backgroundColor = "BRO";
140:         }
141:         else if (index == 3)
142:         {
143:             backgroundColor = "GRA";
144:         }
145:         else if (index == 4)
146:         {
147:             backgroundColor = "GRE";
148:         }
149:         else if (index == 5)
150:         {
151:             backgroundColor = "ORA";
152:         }
153:         else if (index == 6)
154:         {
155:             backgroundColor = "PUR";
156:         }
157:         else if (index == 7)
158:         {
159:             backgroundColor = "RED";
```

```

160:         }
161:         else if (index == 8)
162:         {
163:             backgroundColor = "YEL";
164:         }
165:         else
166:         {
167:             backgroundColor = "BLA";
168:         }
169:
170:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
171:         {
172:             var appSettings = context.AppSettings;
173:             foreach (var appSetting in appSettings)
174:             {
175:                 appSetting.AppBackgroundColor = backgroundColor;
176:             }
177:             context.SubmitChanges();
178:             //CustomMessageBox messageBox = new CustomMessageBox()
179:             //{
180:                 //    Caption = AppResources.BackgroundColor,
181:                 //    Message = AppResources.SuccessfulBackgroundColorChanged,
182:                 //    Background = messageBackGround
183:             //};
184:             //messageBox.Show();
185:             MessageBox.Show(AppResources.BackgroundColorChangedSuccessfully);
186:         }
187:         SetBackgroundColor();
188:         NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
189:     }
190:
191:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
192:     {
193:         if (this.NavigationService.CanGoBack)
194:         {
195:             this.NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
196:         }
197:     }
198:
199:     private void PhoneApplicationPage_Loaded(object sender, RoutedEventArgs e)
200:     {
201:         //SetBackgroundColor();
202:     }
203: }
204: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.13 BilisimSozlugu.cs

This is file BilisimSozlugu.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;

```



```

5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class BilisimSozlugu
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int BilisimSozluguId { get; set; }
19:
20:         [Column]
21:         public string BilisimSozluguName { get; set; }
22:
23:         [Column]
24:         public string BilisimSozluguMeaning { get; set; }
25:
26:         [Column]
27:         public string BilisimSozluguNameMeaning { get; set; }
28:
29:     }
30: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.14 BuyukLugat.cs

This is file BuyukLugat.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class BuyukLugat
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int BuyukLugatId { get; set; }
19:
20:         [Column]
21:         public string BuyukLugatName { get; set; }
22:
23:         [Column]
24:         public string BuyukLugatMeaning { get; set; }
25:
26:         [Column]
27:         public string BuyukLugatNameMeaning { get; set; }

```

```
28:
29:     }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.15 EnglishTurkishVol1.cs

This is file EnglishTurkishVol1.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class EnglishTurkishVol1
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int EnglishVol1Id { get; set; }
19:
20:         [Column]
21:         public string EnglishVol1Name { get; set; }
22:
23:         [Column]
24:         public string EnglishVol1Meaning { get; set; }
25:
26:         [Column]
27:         public string EnglishVol1NameMeaning { get; set; }
28:
29:     }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.16 EnglishTurkishVol2.cs

This is file EnglishTurkishVol2.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
```

```
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class EnglishTurkishVol2
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int EnglishVol2Id { get; set; }
19:
20:         [Column]
21:         public string EnglishVol2Name { get; set; }
22:
23:         [Column]
24:         public string EnglishVol2Meaning { get; set; }
25:
26:         [Column]
27:         public string EnglishVol2NameMeaning { get; set; }
28:
29:     }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.17 Favourite.cs

This is file Favourite.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class Favourite
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int FavouriteId { get; set; }
19:
20:         [Column]
21:         public string FavouriteName { get; set; }
22:
23:         [Column]
24:         public int FavouriteAllId { get; set; }
25:     }
26: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.18 FontFamilySettingsPage.xaml.cs

This is file FontFamilySettingsPage.xaml.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Windows;
7: using System.Windows.Controls;
8: using System.Windows.Controls.Primitives;
9: using System.Windows.Media;
10: using System.Windows.Media.Imaging;
11: using System.Windows.Navigation;
12: using System.Data.Common;
13: using Microsoft.Phone.Controls;
14: using Microsoft.Phone.Controls.Primitives;
15: using Microsoft.Phone.Shell;
16: using AwesomeDictionary.Resources;
17:
18: namespace AwesomeDictionary
19: {
20:     public partial class FontFamilySettingsPage : PhoneApplicationPage
21:     {
22:         public FontFamilySettingsPage()
23:         {
24:             InitializeComponent();
25:
26:             lstFontFamily.Items.Clear();
27:             lstFontFamily.Items.Add("Arial");
28:             lstFontFamily.Items.Add("Arial Black");
29:             lstFontFamily.Items.Add("Baskerville Old Face");
30:             lstFontFamily.Items.Add("Berlin Sans FB");
31:             lstFontFamily.Items.Add("Albumman Old Style");
32:             lstFontFamily.Items.Add("Calibri");
33:             lstFontFamily.Items.Add("Cambria");
34:             lstFontFamily.Items.Add("Candara");
35:             lstFontFamily.Items.Add("Comic Sans MS");
36:             lstFontFamily.Items.Add("Consolas");
37:             lstFontFamily.Items.Add("Constantia");
38:             lstFontFamily.Items.Add("Courier New");
39:             lstFontFamily.Items.Add("DokChampa");
40:             lstFontFamily.Items.Add("Ebrima");
41:             lstFontFamily.Items.Add("Georgia");
42:             lstFontFamily.Items.Add("Lucida Sans Unicode");
43:             lstFontFamily.Items.Add("Meiryo UI");
44:             lstFontFamily.Items.Add("Microsoft YaHei");
45:             lstFontFamily.Items.Add("Malgun Gothic");
46:             lstFontFamily.Items.Add("Segoe UI");
47:             lstFontFamily.Items.Add("Segoe WP");
48:             lstFontFamily.Items.Add("Tahoma");
49:             lstFontFamily.Items.Add("Trebuchet MS");
50:             lstFontFamily.Items.Add("Times New Roman");
51:             lstFontFamily.Items.Add("Verdana");
52:             lstFontFamily.SelectedIndex = -1;
53:

```

```

54:         lblFontFamily.Text = AppResources.SelectFontFamily;
55:         lblGeneralSettings.Text = AppResources.GeneralSettings;
56:
57:         SetBackgroundColor();
58:     }
59:
60:     private void lstFontFamily_SelectionChanged(object sender,
SelectionChangedEventArgs e)
61:     {
62:         if (lstFontFamily.SelectedIndex != -1)
63:         {
64:             using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
65:             {
66:                 var appSettings = context.AppSettings;
67:                 foreach (var item in appSettings)
68:                 {
69:                     item.FontFamily = lstFontFamily.SelectedItem.ToString();
70:                 }
71:                 context.SubmitChanges();
72:                 MessageBox.Show(AppResources.FontFamilyChangedSuccessfully);
73:             }
74:         }
75:         NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
76:     }
77:
78:     protected override void OnNavigatedTo(NavigationEventArgs e)
79:     {
80:         base.OnNavigatedTo(e);
81:     }
82:
83:     protected override void OnNavigatedFrom(NavigationEventArgs e)
84:     {
85:         base.OnNavigatedFrom(e);
86:     }
87:
88:     protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
89:     {
90:         // displays "Fragment: Detail"
91:         //MessageBox.Show("Folder Id: " + e.Fragment);
92:         base.OnFragmentNavigation(e);
93:         //artistId = int.Parse(e.Fragment);
94:         //using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
95:         //{
96:             //    var artist = context.Artists.Where(j =>
j.ArtistId.Equals(artistId)).Single() as Artist;
97:             //    lblArtistName.Text = artist.ArtistName;
98:             //    lblFontFamily.Text = AppResources.SelectFontFamily;
99:         //}
100:
101:     }
102:
103:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
104:     {
105:         if (this.NavigationService.CanGoBack)
106:         {
107:             this.NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
108:         }
109:     }
110:
111:     private void SetBackgroundColor()
112:     {
113:         AppSettings appSettings = new AppSettings();
114:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))

```

```
115:         {
116:             appSettings = context.AppSettings.First() as AppSettings;
117:         }
118:
119:         if (appSettings.AppBackgroundImage != null)
120:         {
121:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
122:             BitmapImage image = new BitmapImage();
123:             image.SetSource(stream);
124:             ImageBrush ib = new ImageBrush();
125:             ib.ImageSource = image;
126:             this.LayoutRoot.Background = ib;
127:         }
128:         else
129:         {
130:             switch (appSettings.AppBackgroundColor)
131:             {
132:                 case "BLA":
133:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
134:                     break;
135:                 case "BLU":
136:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
137:                     break;
138:                 case "BRO":
139:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
140:                     break;
141:                 case "RED":
142:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
143:                     break;
144:                 case "GRE":
145:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
146:                     break;
147:                 case "GRA":
148:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
149:                     break;
150:                 case "YEL":
151:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
152:                     break;
153:                 case "ORA":
154:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
155:                     break;
156:                 case "PUR":
157:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
158:                     break;
159:                 default:
160:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
161:                     break;
162:             }
163:         }
164:     }
165: }
166: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.19 FontSizeSettingsPage.xaml.cs

This is file FontSizeSettingsPage.xaml.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Windows;
7: using System.Windows.Controls;
8: using System.Windows.Controls.Primitives;
9: using System.Windows.Media;
10: using System.Windows.Media.Imaging;
11: using System.Windows.Navigation;
12: using System.Data.Common;
13: using Microsoft.Phone.Controls;
14: using Microsoft.Phone.Controls.Primitives;
15: using Microsoft.Phone.Shell;
16: using AwesomeDictionary.Resources;
17:
18: namespace AwesomeDictionary
19: {
20:     public partial class FontSizeSettingsPage : PhoneApplicationPage
21:     {
22:         public FontSizeSettingsPage()
23:         {
24:             InitializeComponent();
25:
26:             lstFontSize.Items.Clear();
27:             lstFontSize.Items.Add("14");
28:             lstFontSize.Items.Add("18");
29:             lstFontSize.Items.Add("22");
30:             lstFontSize.Items.Add("26");
31:             lstFontSize.Items.Add("28");
32:             lstFontSize.Items.Add("30");
33:             lstFontSize.Items.Add("32");
34:             lstFontSize.Items.Add("34");
35:             lstFontSize.Items.Add("36");
36:             lstFontSize.Items.Add("38");
37:             lstFontSize.Items.Add("40");
38:             lstFontSize.Items.Add("42");
39:             lstFontSize.Items.Add("44");
40:             lstFontSize.Items.Add("64");
41:             lstFontSize.Items.Add("72");
42:             lstFontSize.SelectedIndex = -1;
43:
44:             lblGeneralSettings.Text = AppResources.GeneralSettings;
45:             lblFontSize.Text = AppResources.SelectFontSize;
46:
47:             SetBackgroundColor();
48:         }
49:
50:         private void lstFontSize_SelectionChanged(object sender,
51: SelectionChangedEventArgs e)
52:         {
53:             if (lstFontSize.SelectedIndex != -1)
54:             {
55:                 using (var context = new
56: AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
57:                 {
58:                     var appSettings = context.AppSettings;
59:                     foreach (var item in appSettings)
60:                     {
61:                         item.FontSize = lstFontSize.SelectedItem.ToString();
62:                     }
63:                     context.SubmitChanges();
64:                     MessageBox.Show(AppResources.FontSizeChangedSuccessfully);
65:                 }
66:             }
67:         }
68:     }
69: }

```

```

65:         NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
66:     }
67:
68:     protected override void OnNavigatedTo(NavigationEventArgs e)
69:     {
70:         base.OnNavigatedTo(e);
71:     }
72:
73:     protected override void OnNavigatedFrom(NavigationEventArgs e)
74:     {
75:         base.OnNavigatedFrom(e);
76:     }
77:
78:     protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
79:     {
80:         // displays "Fragment: Detail"
81:         //MessageBox.Show("Folder Id: " + e.Fragment);
82:         base.OnFragmentNavigation(e);
83:         //artistId = int.Parse(e.Fragment);
84:         //using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
85:         //{
86:             //    var artist = context.Artists.Where(j =>
j.ArtistId.Equals(artistId)).Single() as Artist;
87:             //    lblArtistName.Text = artist.ArtistName;
88:             //    lblFontSize.Text = AppResources.SelectFontSize;
89:             //}
90:         }
91:     }
92:
93:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
94:     {
95:         if (this.NavigationService.CanGoBack)
96:         {
97:             this.NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
98:         }
99:     }
100:
101:     private void SetBackgroundColor()
102:     {
103:         AppSettings appSettings = new AppSettings();
104:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
105:         {
106:             appSettings = context.AppSettings.First() as AppSettings;
107:         }
108:
109:         if (appSettings.AppBackgroundImage != null)
110:         {
111:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
112:             BitmapImage image = new BitmapImage();
113:             image.SetSource(stream);
114:             ImageBrush ib = new ImageBrush();
115:             ib.ImageSource = image;
116:             this.LayoutRoot.Background = ib;
117:         }
118:         else
119:         {
120:             switch (appSettings.AppBackgroundColor)
121:             {
122:                 case "BLA":
123:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
124:                     break;
125:                 case "BLU":
126:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
127:                     break;

```



```
128:         case "BRO":
129:             this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
130:             break;
131:         case "RED":
132:             this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
133:             break;
134:         case "GRE":
135:             this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
136:             break;
137:         case "GRA":
138:             this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
139:             break;
140:         case "YEL":
141:             this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
142:             break;
143:         case "ORA":
144:             this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
145:             break;
146:         case "PUR":
147:             this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
148:             break;
149:         default:
150:             this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
151:             break;
152:     }
153: }
154: }
155: }
156: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.20 GeneralSettingsPage.xaml.cs

This is file GeneralSettingsPage.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Globalization;
4: using System.IO;
5: using System.IO.IsolatedStorage;
6: using System.Linq;
7: using System.Net;
8: using System.Text;
9: using System.Threading;
10: using System.Threading.Tasks;
11: using System.Windows;
12: using System.Windows.Controls;
13: using System.Windows.Media;
14: using System.Windows.Media.Imaging;
15: using System.Windows.Navigation;
16: using AwesomeDictionary.Resources;
17: using Microsoft.Phone.Controls;
18: using Microsoft.Phone.Shell;
19: using Microsoft.Phone.Tasks;
20:
21: namespace AwesomeDictionary
```

```

22: {
23:     public partial class GeneralSettingsPage : PhoneApplicationPage
24:     {
25:
26:         public GeneralSettingsPage()
27:         {
28:             InitializeComponent();
29:             SetBackgroundColor();
30:
31:             pvGeneralSettings.Title = AppResources.GeneralSettings;
32:
33:             piLanguage.Header = AppResources.Language;
34:             piDictionaryInstall.Header = AppResources.DictionaryInstall;
35:             //piOtherSettings.Header = AppResources.OtherSettings;
36:             piBackground.Header = AppResources.Background;
37:
38:             //lblOneDrive.Text = AppResources.OneDrive;
39:
40:             btnLanguage.Content = AppResources.Select;
41:             btnBackgroundColor.Content = AppResources.Select;
42:             //btnOneDrive.Content = AppResources.Login;
43:             //btnOneDrive.SignInText = AppResources.SignIn;
44:             //btnOneDrive.SignOutText = AppResources.SignOut;
45:             txtInstalling.Text = AppResources.Installing;
46:
47:             pbInstall.Visibility = Visibility.Collapsed;
48:             txtInstalling.Visibility = Visibility.Collapsed;
49:             txtInstalling.BorderBrush = this.LayoutRoot.Background;
50:
51:             btnRemoveBackgroundImage.Content = AppResources.RemoveBackgroundImage;
52:             lblBackgroundImage.Text = AppResources.BackgroundImage;
53:             btnBackgroundImage.Content = AppResources.Select;
54:             btnResetSettings.Content = AppResources.ResetSettings;
55:
56:             piFont.Header = AppResources.Font;
57:             btnFontFamily.Content = AppResources.Select;
58:             btnFontSize.Content = AppResources.Select;
59:
60:             btnInstall.Content = AppResources.Install;
61:             btnUninstall.Content = AppResources.Uninstall;
62:
63:             txtBuyukLugat.Text = AppResources.BuyukLugat + " (49331 " +
AppResources.Word + ")";
64:             txtComputer.Text = AppResources.ComputerDictionary + " (3508 " +
AppResources.Word + ")";
65:             txtGerman.Text = AppResources.GermanTurkish + " (17526 " +
AppResources.Word + ")";
66:             txtOxford.Text = AppResources.OxfordDictionary + " (36369 " +
AppResources.Word + ")";
67:             txtWordMeaning.Text = AppResources.WordMeaning + " (10535 " +
AppResources.Word + ")";
68:             txtRisaleNur.Text = AppResources.RisaleNur + " (9478 " +
AppResources.Word + ")";
69:             txtEnglishVol1.Text = AppResources.EnglishTurkishVol1 + " (127157 " +
AppResources.Word + ")";
70:             txtEnglishVol2.Text = AppResources.EnglishTurkishVol2 + " (3699 " +
AppResources.Word + ")";
71:
72:
73:             //cbSync.Content = AppResources.SyncOnOneFile;
74:             //cbSync.IsEnabled = false;
75:
76:             SetBackgroundColor();
77:
78:             using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
79:             {
80:                 var appSettings = context.AppSettings.First() as AppSettings;
81:

```

```

82:         lblFontFamily.Text = AppResources.FontFamily + " (" +
AppResources.Selected + ": " + appSettings.FontFamily + ")";
83:         lblFontSize.Text = AppResources.FontSize + " (" +
AppResources.Selected + ": " + appSettings.FontSize + ")";
84:
85:         if (appSettings.AppLangName == "EN")
86:         {
87:             lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.English + ")";
88:         }
89:         if (appSettings.AppLangName == "TR")
90:         {
91:             lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Turkish + ")";
92:         }
93:         if (appSettings.AppLangName == "DE")
94:         {
95:             lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.German + ")";
96:         }
97:         //if (appSettings.AppLangName == "ES")
98:         //{
99:             //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Spanish + ")";
100:        //}
101:
102:        //if (appSettings.AppLangName == "PT")
103:        //{
104:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Portuguese + ")";
105:        //}
106:        //if (appSettings.AppLangName == "AR")
107:        //{
108:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Arabic + ")";
109:        //}
110:        //if (appSettings.AppLangName == "FA")
111:        //{
112:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Persian + ")";
113:        //}
114:        //if (appSettings.AppLangName == "IT")
115:        //{
116:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Italian + ")";
117:        //}
118:        //if (appSettings.AppLangName == "FR")
119:        //{
120:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.French + ")";
121:        //}
122:        //if (appSettings.AppLangName == "RU")
123:        //{
124:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Russian + ")";
125:        //}
126:        //if (appSettings.AppLangName == "ZH")
127:        //{
128:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Chinese + ")";
129:        //}
130:        //if (appSettings.AppLangName == "JA")
131:        //{
132:            //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Japanese + ")";
133:        //}
134:        //if (appSettings.AppLangName == "SA")
135:        //{
136:            //    lblLanguage.Text = AppResources.Language + " (" +

```

```

AppResources.Selected + ": " + AppResources.Sanskrit + "));
137:         //}
138:         //if (appSettings.AppLangName == "TH")
139:         //{
140:         //    lblLanguage.Text = AppResources.Language + " (" +
AppResources.Selected + ": " + AppResources.Thai + "));
141:         //}
142:
143:         if (appSettings.AppBackgroundColor == "BLA")
144:         {
145:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Black + "));
146:         }
147:         if (appSettings.AppBackgroundColor == "BLU")
148:         {
149:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Blue + "));
150:         }
151:         if (appSettings.AppBackgroundColor == "BRO")
152:         {
153:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Brown + "));
154:         }
155:         if (appSettings.AppBackgroundColor == "RED")
156:         {
157:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Red + "));
158:         }
159:         if (appSettings.AppBackgroundColor == "GRE")
160:         {
161:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Green + "));
162:         }
163:         if (appSettings.AppBackgroundColor == "YEL")
164:         {
165:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Yellow + "));
166:         }
167:         if (appSettings.AppBackgroundColor == "GRA")
168:         {
169:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Gray + "));
170:         }
171:         if (appSettings.AppBackgroundColor == "ORA")
172:         {
173:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Orange + "));
174:         }
175:         if (appSettings.AppBackgroundColor == "PUR")
176:         {
177:             lblBackgroundColor.Text = AppResources.BackgroundColor + " (" +
AppResources.Selected + ": " + AppResources.Purple + "));
178:         }
179:     }
180: }
181:
182: protected override void OnNavigatedTo(NavigationEventArgs e)
183: {
184:     base.OnNavigatedTo(e);
185:     SetBackgroundColor();
186:     //while (NavigationService.CanGoBack)
187:     //NavigationService.RemoveBackEntry();
188: }
189:
190:
191: protected override void OnNavigatedFrom(NavigationEventArgs e)
192: {
193:     base.OnNavigatedFrom(e);
194:     //while (NavigationService.CanGoBack)

```

```

195:         //NavigationService.RemoveBackEntry();
196:
197:     }
198:
199:     private void SetBackgroundColor()
200:     {
201:         AppSettings appSettings = new AppSettings();
202:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
203:         {
204:             appSettings = context.AppSettings.First() as AppSettings;
205:         }
206:
207:         if (appSettings.AppBackgroundImage != null)
208:         {
209:             MemoryStream stream = new
MemoryStream(appSettings.AppBackgroundImage);
210:             BitmapImage image = new BitmapImage();
211:             image.SetSource(stream);
212:             ImageBrush ib = new ImageBrush();
213:             ib.ImageSource = image;
214:             this.LayoutRoot.Background = ib;
215:         }
216:         else
217:         {
218:             switch (appSettings.AppBackgroundColor)
219:             {
220:                 case "BLA":
221:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Black);
222:                     break;
223:                 case "BLU":
224:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
225:                     break;
226:                 case "BRO":
227:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Brown);
228:                     break;
229:                 case "RED":
230:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
231:                     break;
232:                 case "GRE":
233:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Green);
234:                     break;
235:                 case "GRA":
236:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
237:                     break;
238:                 case "YEL":
239:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
240:                     break;
241:                 case "ORA":
242:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
243:                     break;
244:                 case "PUR":
245:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
246:                     break;
247:                 default:
248:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Black);
249:                     break;
250:             }
251:         }
252:     }
253:
254:     private void btnLanguage_Click(object sender, RoutedEventArgs e)

```

```

255:         {
256:             this.NavigationService.Navigate(new Uri("/LanguageSettingsPage.xaml",
UriKind.Relative));
257:         }
258:
259:         private void btnBackgroundColor_Click(object sender, RoutedEventArgs e)
260:         {
261:             NavigationService.Navigate(new Uri("/BackgroundColorSettingsPage.xaml",
UriKind.Relative));
262:         }
263:
264:         private void btnBackgroundImage_Click(object sender, RoutedEventArgs e)
265:         {
266:             PhotoChooserTask objPhotoChooser = new PhotoChooserTask();
267:             objPhotoChooser.Completed += new
EventHandler<PhotoResult>(PhotoChooseCall);
268:             objPhotoChooser.Show();
269:         }
270:
271:         private void PhotoChooseCall(object sender, PhotoResult e)
272:         {
273:             switch (e.TaskResult)
274:             {
275:                 case TaskResult.OK:
276:                     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
277:                     {
278:                         var appSettings = context.AppSettings;
279:                         foreach (var appSetting in appSettings)
280:                         {
281:                             appSetting.AppBackgroundImage = new
byte[e.ChosenPhoto.Length];
282:                             e.ChosenPhoto.Position = 0;
283:                             e.ChosenPhoto.Read(appSetting.AppBackgroundImage, 0,
appSetting.AppBackgroundImage.Length);
284:                             //noteFolder.NoteFolderPassword = "";
285:                         }
286:                         context.SubmitChanges();
287:                         MessageBox.Show(AppResources.BackgroundImageChangedSuccessfully);
288:                     }
289:                     break;
290:                 case TaskResult.Cancel:
291:                     //MessageBox.Show("Cancelled");
292:                     break;
293:                 case TaskResult.None:
294:                     //MessageBox.Show("Nothing Entered");
295:                     break;
296:             }
297:             SetBackgroundColor();
298:         }
299:
300:         private void btnRemoveBackgroundImage_Click(object sender, RoutedEventArgs e)
301:         {
302:             using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
303:             {
304:                 var appSettings = context.AppSettings;
305:                 foreach (var appSetting in appSettings)
306:                 {
307:                     appSetting.AppBackgroundImage = null;
308:                 }
309:                 context.SubmitChanges();
310:                 MessageBox.Show(AppResources.BackgroundImageRemovedSuccessfully);
311:             }
312:         }
313:
314:         private void PhoneApplicationPage_Loaded(object sender, RoutedEventArgs e)
315:         {

```

```

316:
317:         //pvGeneralSettings.Title = AppResources.GeneralSettings;
318:
319:         //piLanguage.Header = AppResources.Language;
320:         //piSync.Header = AppResources.Sync;
321:         //piOtherSettings.Header = AppResources.OtherSettings;
322:         //piBackground.Header = AppResources.Background;
323:
324:         ///lblOneDrive.Text = AppResources.OneDrive;
325:
326:         //btnCategoryOrder.Content = AppResources.Select;
327:         //btnCategoryOrderStyle.Content = AppResources.Select;
328:         //btnLanguage.Content = AppResources.Select;
329:         //btnBackgroundColor.Content = AppResources.Select;
330:         ///btnOneDrive.Content = AppResources.Login;
331:         ///btnOneDrive.SignInText = AppResources.SignIn;
332:         ///btnOneDrive.SignOutText = AppResources.SignOut;
333:         //btnOneDriveSync.Content = AppResources.Sync;
334:         //lblOneDrive.Text = AppResources.OneDrive;
335:         //txtInstalling.Text = AppResources.Synchronizing;
336:
337:         //pbSync.Visibility = Visibility.Collapsed;
338:         //txtInstalling.Visibility = Visibility.Collapsed;
339:         //txtInstalling.BorderBrush = this.LayoutRoot.Background;
340:
341:         //btnRemoveBackgroundImage.Content = AppResources.RemoveBackgroundImage;
342:         //lblBackgroundImage.Text = AppResources.BackgroundImage;
343:         //btnBackgroundImage.Content = AppResources.Select;
344:         //btnResetSettings.Content = AppResources.ResetSettings;
345:
346:         //btnOneDriveSync.IsEnabled = false;
347:         //cbSync.Content = AppResources.SyncOnOneFile;
348:         //cbSync.IsEnabled = false;
349:         //btnOneDrive.Content = "Sign In";
350:
351:         //SetBackgroundColor();
352:     }
353:
354:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
355:     {
356:         if (this.NavigationService.CanGoBack)
357:         {
358:             this.NavigationService.Navigate(new Uri("/MainPage.xaml",
UriKind.Relative));
359:         }
360:     }
361:
362:     private void btnResetSettings_Click(object sender, RoutedEventArgs e)
363:     {
364:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
365:         {
366:             var appSettings = context.AppSettings;
367:             foreach (var appSetting in appSettings)
368:             {
369:                 appSetting.AppBackgroundImage = null;
370:                 appSetting.AppBackgroundColor = "BLA";
371:             }
372:             context.SubmitChanges();
373:             this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
374:             MessageBox.Show(AppResources.BackgroundSettingsResetSuccessfully);
375:         }
376:     }
377:
378:     private void btnFontSize_Click(object sender, RoutedEventArgs e)
379:     {
380:         this.NavigationService.Navigate(new Uri("/FontSizeSettingsPage.xaml",
UriKind.Relative));

```



```

381:         }
382:
383:         private void btnFontFamily_Click(object sender, RoutedEventArgs e)
384:         {
385:             this.NavigationService.Navigate(new Uri("/FontFamilySettingsPage.xaml",
UriKind.Relative));
386:         }
387:
388:         // burasi düzenlenecek
389:         private void btnInstall_Click(object sender, RoutedEventArgs e)
390:         {
391:             try
392:             {
393:
394:                 if (cbBuyukLugat.IsChecked == false && cbComputer.IsChecked == false
&& cbEnglishVoll.IsChecked == false && cbEnglishVol2.IsChecked == false &&
cbOxford.IsChecked == false && cbRisaleNur.IsChecked == false && cbWordMeaning.IsChecked ==
false && cbGerman.IsChecked == false)
395:                 {
396:                     MessageBox.Show(AppResources.AtLeastOneDictionary);
397:                 }
398:                 else
399:                 {
400:                     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
401:                     {
402:                         SozlukYukle(context);
403:                     }
404:                 }
405:             }
406:             catch (Exception ex)
407:             {
408:                 MessageBox.Show(AppResources.SystemFault);
409:             }
410:         }
411:
412:         private void SozlukYukle2(AwesomeDictionaryDataContext context)
413:         {
414:             if (cbBuyukLugat.IsChecked == true)
415:             {
416:                 if (context.BuyukLugats.ToList().Count > 0)
417:                 {
418:                     var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.BuyukLugat)).ToList() as List<All>;
419:                     context.AllNames.DeleteAllOnSubmit(allList);
420:
context.BuyukLugats.DeleteAllOnSubmit(context.BuyukLugats.ToList());
421:                 }
422:                 //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
423:                 var buyukLugat = ReadFile(@"Sources\BuyukLugatEsas.txt");
424:                 var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
425:                 for (int i = 1; i < buyukLugatLines.Count; i++)
426:                 {
427:                     var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
428:                     BuyukLugat buyukLugatWord = new BuyukLugat();
429:                     buyukLugatWord.BuyukLugatName =
information[0].ToUpper().TrimEnd().TrimStart();
430:                     buyukLugatWord.BuyukLugatMeaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
431:                     buyukLugatWord.BuyukLugatNameMeaning =
buyukLugatWord.BuyukLugatName + " " + buyukLugatWord.BuyukLugatNameMeaning;
432:                     context.BuyukLugats.InsertOnSubmit(buyukLugatWord);
433:                 }
434:                 context.SubmitChanges();
435:                 var buyukLugats = context.BuyukLugats.ToList() as List<BuyukLugat>;
436:                 for (int i = 0; i < buyukLugats.Count; i++)
437:                 {

```



```

438:         All allName = new All();
439:         allName.AllName = buyukLugats[i].BuyukLugatName;
440:         allName.AllMeaning = buyukLugats[i].BuyukLugatMeaning;
441:         allName.AllNameMeaning = buyukLugats[i].BuyukLugatName + " " +
buyukLugats[i].BuyukLugatMeaning;
442:         allName.AllSource = AppResources.BuyukLugat;
443:         allName.AllNameSource = allName.AllName + " (" +
AppResources.BuyukLugat + ")";
444:         context.AllNames.InsertOnSubmit(allName);
445:     }
446:     context.SubmitChanges();
447: }
448:
449:     if (cbGerman.IsChecked == true)
450:     {
451:         if (context.AlmancaTurkces.ToList().Count > 0)
452:         {
453:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.GermanTurkish)).ToList() as List<All>;
454:             context.AllNames.DeleteAllOnSubmit(allList);
455:
context.AlmancaTurkces.DeleteAllOnSubmit(context.AlmancaTurkces.ToList());
456:         }
457:         //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
458:         var almancaTurkce = ReadFile(@"Sources\AlmancaTurkceSozluk.txt");
459:         var almancaTurkceLines = almancaTurkce.Split('\n').ToList() as
List<string>;
460:         for (int i = 1; i < almancaTurkceLines.Count; i++)
461:         {
462:             var information = almancaTurkceLines[i].Split('_').ToList() as
List<string>;
463:             AlmancaTurkce almancaTurkceWord = new AlmancaTurkce();
464:             almancaTurkceWord.AlmancaTurkceName =
information[0].ToUpper().TrimEnd().TrimStart();
465:             almancaTurkceWord.AlmancaTurkceMeaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
466:             almancaTurkceWord.AlmancaTurkceNameMeaning =
almancaTurkceWord.AlmancaTurkceName + " " + almancaTurkceWord.AlmancaTurkceNameMeaning;
467:             context.AlmancaTurkces.InsertOnSubmit(almancaTurkceWord);
468:         }
469:         context.SubmitChanges();
470:         var almancaTurkces = context.AlmancaTurkces.ToList() as
List<AlmancaTurkce>;
471:         for (int i = 0; i < almancaTurkces.Count; i++)
472:         {
473:             All allName = new All();
474:             allName.AllName = almancaTurkces[i].AlmancaTurkceName;
475:             allName.AllMeaning = almancaTurkces[i].AlmancaTurkceMeaning;
476:             allName.AllNameMeaning = almancaTurkces[i].AlmancaTurkceName + "
" + almancaTurkces[i].AlmancaTurkceMeaning;
477:             allName.AllSource = AppResources.GermanTurkish;
478:             allName.AllNameSource = allName.AllName + " (" +
AppResources.GermanTurkish + ")";
479:             context.AllNames.InsertOnSubmit(allName);
480:         }
481:         context.SubmitChanges();
482:     }
483:
484:     if (cbComputer.IsChecked == true)
485:     {
486:         if (context.Bilisims.ToList().Count > 0)
487:         {
488:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.ComputerDictionary)).ToList() as List<All>;
489:             context.AllNames.DeleteAllOnSubmit(allList);
490:             context.Bilisims.DeleteAllOnSubmit(context.Bilisims.ToList());
491:         }
492:         //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme

```

```

yapiliyor
493:         var bilisimSozlugu = ReadFile(@"Sources\BilisimSozlugu.txt");
494:         var bilisimSozluguLines = bilisimSozlugu.Split('\n').ToList() as
List<string>;
495:         for (int i = 1; i < bilisimSozluguLines.Count; i++)
496:         {
497:             var information = bilisimSozluguLines[i].Split('_').ToList() as
List<string>;
498:             BilisimSozlugu bilisimSozluguWord = new BilisimSozlugu();
499:             bilisimSozluguWord.BilisimSozluguName =
information[0].ToUpper().TrimEnd().TrimStart();
500:             bilisimSozluguWord.BilisimSozluguMeaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
501:             bilisimSozluguWord.BilisimSozluguNameMeaning =
bilisimSozluguWord.BilisimSozluguName + " " + bilisimSozluguWord.BilisimSozluguNameMeaning;
502:             context.Bilisims.InsertOnSubmit(bilisimSozluguWord);
503:         }
504:         context.SubmitChanges();
505:         var bilisimSozlugus = context.Bilisims.ToList() as
List<BilisimSozlugu>;
506:         for (int i = 0; i < bilisimSozlugus.Count; i++)
507:         {
508:             All allName = new All();
509:             allName.AllName = bilisimSozlugus[i].BilisimSozluguName;
510:             allName.AllMeaning = bilisimSozlugus[i].BilisimSozluguMeaning;
511:             allName.AllNameMeaning = bilisimSozlugus[i].BilisimSozluguName +
" " + bilisimSozlugus[i].BilisimSozluguMeaning;
512:             allName.AllSource = AppResources.ComputerDictionary;
513:             allName.AllNameSource = allName.AllName + " (" +
AppResources.ComputerDictionary + ")";
514:             context.AllNames.InsertOnSubmit(allName);
515:         }
516:         context.SubmitChanges();
517:     }
518:
519:     if (cbOxford.IsChecked == true)
520:     {
521:         if (context.Oxfords.ToList().Count > 0)
522:         {
523:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.OxfordDictionary)).ToList() as List<All>;
524:             context.AllNames.DeleteAllOnSubmit(allList);
525:             context.Oxfords.DeleteAllOnSubmit(context.Oxfords.ToList());
526:         }
527:         //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
528:         var oxfordEnglishEnglish =
ReadFile(@"Sources\OxfordEnglishTurkish.txt");
529:         var oxfordEnglishEnglishLines =
oxfordEnglishEnglish.Split('\n').ToList() as List<string>;
530:         for (int i = 1; i < oxfordEnglishEnglishLines.Count; i++)
531:         {
532:             var information =
oxfordEnglishEnglishLines[i].Split('_').ToList() as List<string>;
533:             OxfordEnglishEnglish oxfordEnglishEnglishWord = new
OxfordEnglishEnglish();
534:             oxfordEnglishEnglishWord.OxfordName =
information[0].ToUpper().TrimEnd().TrimStart();
535:             oxfordEnglishEnglishWord.OxfordMeaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
536:             oxfordEnglishEnglishWord.OxfordNameMeaning =
oxfordEnglishEnglishWord.OxfordName + " " + oxfordEnglishEnglishWord.OxfordMeaning;
537:             context.Oxfords.InsertOnSubmit(oxfordEnglishEnglishWord);
538:         }
539:         context.SubmitChanges();
540:         var oxfordEnglishEnglishs = context.Oxfords.ToList() as
List<OxfordEnglishEnglish>;
541:         for (int i = 0; i < oxfordEnglishEnglishs.Count; i++)
542:         {

```

```

543:         All allName = new All();
544:         allName.AllName = oxfordEnglishEnglisms[i].OxfordName;
545:         allName.AllMeaning = oxfordEnglishEnglisms[i].OxfordMeaning;
546:         allName.AllNameMeaning = oxfordEnglishEnglisms[i].OxfordName + "
" + oxfordEnglishEnglisms[i].OxfordMeaning;
547:         allName.AllSource = AppResources.OxfordDictionary;
548:         allName.AllNameSource = allName.AllName + " (" +
AppResources.OxfordDictionary + ")";
549:         context.AllNames.InsertOnSubmit(allName);
550:     }
551:     context.SubmitChanges();
552: }
553:
554:     if (cbRisaleNur.IsChecked == true)
555:     {
556:         if (context.RisaleNurs.ToList().Count > 0)
557:         {
558:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.RisaleNur)).ToList() as List<All>;
559:             context.AllNames.DeleteAllOnSubmit(allList);
560:             context.RisaleNurs.DeleteAllOnSubmit(context.RisaleNurs.ToList());
561:         }
562:         //burada metin belgeleri okunuyor ve veritabanındaki tablolara ekleme
yapiliyor
563:         var risaleNur = ReadFile(@"Sources\RisaleNurSozlugu.txt");
564:         var risaleNurLines = risaleNur.Split('\n').ToList() as List<string>;
565:         for (int i = 1; i < risaleNurLines.Count; i++)
566:         {
567:             var information = risaleNurLines[i].Split('_').ToList() as
List<string>;
568:             RisaleNur risaleNurWord = new RisaleNur();
569:             risaleNurWord.RisaleNurName =
information[0].ToUpper().TrimEnd().TrimStart();
570:             risaleNurWord.RisaleNurMeaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
571:             risaleNurWord.RisaleNurNameMeaning = risaleNurWord.RisaleNurName
+ " " + risaleNurWord.RisaleNurNameMeaning;
572:             context.RisaleNurs.InsertOnSubmit(risaleNurWord);
573:         }
574:         context.SubmitChanges();
575:         var risaleNurs = context.RisaleNurs.ToList() as List<RisaleNur>;
576:         for (int i = 0; i < risaleNurs.Count; i++)
577:         {
578:             All allName = new All();
579:             allName.AllName = risaleNurs[i].RisaleNurName;
580:             allName.AllMeaning = risaleNurs[i].RisaleNurMeaning;
581:             allName.AllNameMeaning = risaleNurs[i].RisaleNurName + " " +
risaleNurs[i].RisaleNurMeaning;
582:             allName.AllSource = AppResources.RisaleNur;
583:             allName.AllNameSource = allName.AllName + " (" +
AppResources.RisaleNur + ")";
584:             context.AllNames.InsertOnSubmit(allName);
585:         }
586:     }
587:     context.SubmitChanges();
588: }
589:
590:     if (cbEnglishVoll1.IsChecked == true)
591:     {
592:         if (context.EnglishTurkishVolls.ToList().Count > 0)
593:         {
594:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.EnglishTurkishVoll1)).ToList() as List<All>;
595:             context.AllNames.DeleteAllOnSubmit(allList);
596:         }
597:     }
598:     context.EnglishTurkishVolls.DeleteAllOnSubmit(context.EnglishTurkishVolls.ToList());
599: }
600: //burada metin belgeleri okunuyor ve veritabanındaki tablolara ekleme
yapiliyor

```

```

599:         var englishTurkishVol1 =
ReadFile(@"Sources\IngilizceTürkçeSozlukv1.txt");
600:         var englishTurkishVol1Lines = englishTurkishVol1.Split('\n').ToList()
as List<string>;
601:         for (int i = 1; i < englishTurkishVol1Lines.Count; i++)
602:         {
603:             var information = englishTurkishVol1Lines[i].Split('_').ToList()
as List<string>;
604:             EnglishTurkishVol1 englishTurkishVol1Word = new
EnglishTurkishVol1();
605:             englishTurkishVol1Word.EnglishVol1Name =
information[0].ToUpper().TrimEnd().TrimStart();
606:             englishTurkishVol1Word.EnglishVol1Meaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
607:             englishTurkishVol1Word.EnglishVol1NameMeaning =
englishTurkishVol1Word.EnglishVol1Name + " " +
englishTurkishVol1Word.EnglishVol1NameMeaning;
608:             context.EnglishTurkishVolls.InsertOnSubmit(englishTurkishVol1Word);
609:         }
610:         context.SubmitChanges();
611:         var englishTurkishVolls = context.EnglishTurkishVolls.ToList() as
List<EnglishTurkishVol1>;
612:         for (int i = 0; i < englishTurkishVolls.Count; i++)
613:         {
614:             All allName = new All();
615:             allName.AllName = englishTurkishVolls[i].EnglishVol1Name;
616:             allName.AllMeaning = englishTurkishVolls[i].EnglishVol1Meaning;
617:             allName.AllNameMeaning = englishTurkishVolls[i].EnglishVol1Name +
" " + englishTurkishVolls[i].EnglishVol1NameMeaning;
618:             allName.AllSource = AppResources.EnglishTurkishVol1;
619:             allName.AllNameSource = allName.AllName + " (" +
AppResources.EnglishTurkishVol1 + ")";
620:             context.AllNames.InsertOnSubmit(allName);
621:         }
622:         context.SubmitChanges();
623:     }
624:
625:     if (cbEnglishVol2.IsChecked == true)
626:     {
627:         if (context.EnglishTurkishVol2s.ToList().Count > 0)
628:         {
629:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.EnglishTurkishVol2)).ToList() as List<All>;
630:             context.AllNames.DeleteAllOnSubmit(allList);
631:
632:             context.EnglishTurkishVol2s.DeleteAllOnSubmit(context.EnglishTurkishVol2s.ToList());
633:             //burada metin belgeleri okunuyor ve veritabanındaki tablolara ekleme
yapiliyor
634:             var englishTurkishVol2 =
ReadFile(@"Sources\IngilizceTürkçeSozlukv2.txt");
635:             var englishTurkishVol2Lines = englishTurkishVol2.Split('\n').ToList()
as List<string>;
636:             for (int i = 1; i < englishTurkishVol2Lines.Count; i++)
637:             {
638:                 var information = englishTurkishVol2Lines[i].Split('_').ToList()
as List<string>;
639:                 EnglishTurkishVol2 englishTurkishVol2Word = new
EnglishTurkishVol2();
640:                 englishTurkishVol2Word.EnglishVol2Name =
information[0].ToUpper().TrimEnd().TrimStart();
641:                 englishTurkishVol2Word.EnglishVol2Meaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
642:                 englishTurkishVol2Word.EnglishVol2NameMeaning =
englishTurkishVol2Word.EnglishVol2Name + " " +
englishTurkishVol2Word.EnglishVol2NameMeaning;
643:                 context.EnglishTurkishVol2s.InsertOnSubmit(englishTurkishVol2Word);

```

```

644:         }
645:         context.SubmitChanges();
646:         var englishTurkishVol2s = context.EnglishTurkishVol2s.ToList() as
List<EnglishTurkishVol2>;
647:         for (int i = 0; i < englishTurkishVol2s.Count; i++)
648:         {
649:             All allName = new All();
650:             allName.AllName = englishTurkishVol2s[i].EnglishVol2Name;
651:             allName.AllMeaning = englishTurkishVol2s[i].EnglishVol2Meaning;
652:             allName.AllNameMeaning = englishTurkishVol2s[i].EnglishVol2Name +
" " + englishTurkishVol2s[i].EnglishVol2NameMeaning;
653:             allName.AllSource = AppResources.EnglishTurkishVol2;
654:             allName.AllNameSource = allName.AllName + " (" +
AppResources.EnglishTurkishVol2 + ")";
655:             context.AllNames.InsertOnSubmit(allName);
656:         }
657:         context.SubmitChanges();
658:     }
659:
660:     if (cbWordMeaning.IsChecked == true)
661:     {
662:         if (context.Kelimes.ToList().Count > 0)
663:         {
664:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.WordMeaning)).ToList() as List<All>;
665:             context.AllNames.DeleteAllOnSubmit(allList);
666:             context.Kelimes.DeleteAllOnSubmit(context.Kelimes.ToList());
667:         }
668:         //burada metin belgeleri okunuyor ve veritabanındaki tablolara ekleme
yapiliyor
669:         var kelimeAnlamlari = ReadFile(@"Sources\KelimeAnlamlari.txt");
670:         var kelimeAnlamlariLines = kelimeAnlamlari.Split('\n').ToList() as
List<string>;
671:         for (int i = 1; i < kelimeAnlamlariLines.Count; i++)
672:         {
673:             var information = kelimeAnlamlariLines[i].Split('_').ToList() as
List<string>;
674:             KelimeAnlamlari kelimeAnlamlariWord = new KelimeAnlamlari();
675:             kelimeAnlamlariWord.KelimeAnlamlariName =
information[0].ToUpper().TrimEnd().TrimStart();
676:             kelimeAnlamlariWord.KelimeAnlamlariMeaning =
information[1].TrimEnd().TrimStart() == null ? "" : information[1].TrimEnd().TrimStart();
677:             kelimeAnlamlariWord.KelimeAnlamlariNameMeaning =
kelimeAnlamlariWord.KelimeAnlamlariName + " " +
kelimeAnlamlariWord.KelimeAnlamlariNameMeaning;
678:             context.Kelimes.InsertOnSubmit(kelimeAnlamlariWord);
679:         }
680:         context.SubmitChanges();
681:         var kelimeAnlamlaris = context.Kelimes.ToList() as
List<KelimeAnlamlari>;
682:         for (int i = 0; i < kelimeAnlamlaris.Count; i++)
683:         {
684:             All allName = new All();
685:             allName.AllName = kelimeAnlamlaris[i].KelimeAnlamlariName;
686:             allName.AllMeaning = kelimeAnlamlaris[i].KelimeAnlamlariMeaning;
687:             allName.AllNameMeaning = kelimeAnlamlaris[i].KelimeAnlamlariName
+ " " + kelimeAnlamlaris[i].KelimeAnlamlariNameMeaning;
688:             allName.AllSource = AppResources.WordMeaning;
689:             allName.AllNameSource = allName.AllName + " (" +
AppResources.WordMeaning + ")";
690:             context.AllNames.InsertOnSubmit(allName);
691:         }
692:         context.SubmitChanges();
693:     }
694:
695:     MessageBox.Show(AppResources.DictionariesInstalledSuccessfully);
696: }
697:
698: private void SozlukYukle(AwesomeDictionaryDataContext context)

```

```

699:         {
700:             //pbInstall.Visibility = Visibility.Visible;
701:
702:             if (cbBuyukLugat.IsChecked == true)
703:             {
704:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.BuyukLugat)).ToList() as List<All>;
705:                 if (allList != null)
706:                 {
707:                     context.AllNames.DeleteAllOnSubmit(allList);
708:                 }
709:                 //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
710:                 var buyukLugat = ReadFile(@"Sources\BuyukLugatEsas.txt");
711:                 var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
712:                 string text = AppResources.Installing + " " + AppResources.BuyukLugat;
713:                 //pbInstall.Value = 0;
714:                 List<All> allNames = new List<All>();
715:                 for (int i = 1; i < buyukLugatLines.Count; i++)
716:                 {
717:                     //pbInstall.Value = i;
718:                     var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
719:                     All allName = new All();
720:                     allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
721:                     allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? " " : information[1].TrimEnd().TrimStart();
722:                     allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
723:                     allName.AllSource = AppResources.BuyukLugat;
724:                     allName.AllNameSource = allName.AllName + " (" +
AppResources.BuyukLugat + ")";
725:                     allNames.Add(allName);
726:                     //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
727:                 }
728:                 context.AllNames.InsertAllOnSubmit(allNames);
729:             }
730:
731:             if (cbGerman.IsChecked == true)
732:             {
733:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.GermanTurkish)).ToList() as List<All>;
734:                 if (allList != null)
735:                 {
736:                     context.AllNames.DeleteAllOnSubmit(allList);
737:                 }
738:                 //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
739:                 var buyukLugat = ReadFile(@"Sources\AlmancaTurkceSozluk.txt");
740:                 var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
741:                 string text = AppResources.Installing + " " +
AppResources.GermanTurkish;
742:                 //pbInstall.Value = 0;
743:                 List<All> allNames = new List<All>();
744:                 for (int i = 1; i < buyukLugatLines.Count; i++)
745:                 {
746:                     //pbInstall.Value = i;
747:                     var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
748:                     All allName = new All();
749:                     allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
750:                     allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? " " : information[1].TrimEnd().TrimStart();
751:                     allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
752:                     allName.AllSource = AppResources.GermanTurkish;
753:                     allName.AllNameSource = allName.AllName + " (" +
AppResources.GermanTurkish + ")";

```



```

754:             allNames.Add(allName);
755:             //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
756:         }
757:         context.AllNames.InsertAllOnSubmit(allNames);
758:     }
759: }
760:
761:         if (cbComputer.IsChecked == true)
762:         {
763:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.ComputerDictionary)).ToList() as List<All>;
764:             if (allList != null)
765:             {
766:                 context.AllNames.DeleteAllOnSubmit(allList);
767:             }
768:             //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
769:             var buyukLugat = ReadFile(@"Sources\BilisimSozlugu.txt");
770:             var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
771:             string text = AppResources.Installing + " " +
AppResources.ComputerDictionary;
772:             //pbInstall.Value = 0;
773:             List<All> allNames = new List<All>();
774:             for (int i = 1; i < buyukLugatLines.Count; i++)
775:             {
776:                 //pbInstall.Value = i;
777:                 var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
778:                 All allName = new All();
779:                 allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
780:                 allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? "" : information[1].TrimEnd().TrimStart();
781:                 allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
782:                 allName.AllSource = AppResources.ComputerDictionary;
783:                 allName.AllNameSource = allName.AllName + " (" +
AppResources.ComputerDictionary + ")";
784:                 allNames.Add(allName);
785:                 //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
786:             }
787:             context.AllNames.InsertAllOnSubmit(allNames);
788:         }
789:     }
790:     if (cbOxford.IsChecked == true)
791:     {
792:         var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.OxfordDictionary)).ToList() as List<All>;
793:         if (allList != null)
794:         {
795:             context.AllNames.DeleteAllOnSubmit(allList);
796:         }
797:         //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
798:         var buyukLugat = ReadFile(@"Sources\OxfordEnglishEnglish.txt");
799:         var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
800:         string text = AppResources.Installing + " " +
AppResources.OxfordDictionary;
801:         //pbInstall.Value = 0;
802:         List<All> allNames = new List<All>();
803:         for (int i = 1; i < buyukLugatLines.Count; i++)
804:         {
805:             //pbInstall.Value = i;
806:             var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
807:             All allName = new All();
808:             allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
809:             allName.AllMeaning = information[1].TrimEnd().TrimStart() == null

```

```

? "" : information[1].TrimEnd().TrimStart();
810:         allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
811:         allName.AllSource = AppResources.OxfordDictionary;
812:         allName.AllNameSource = allName.AllName + " (" +
AppResources.OxfordDictionary + ")";
813:         allNames.Add(allName);
814:         //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
815:     }
816:     context.AllNames.InsertAllOnSubmit(allNames);
817: }
818:
819:     if (cbRisaleNur.IsChecked == true)
820:     {
821:         var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.RisaleNur)).ToList() as List<All>;
822:         if (allList != null)
823:         {
824:             context.AllNames.DeleteAllOnSubmit(allList);
825:         }
826:         //burada metin belgeleri okunuyor ve veritabanındaki tablolara ekleme
yapiliyor
827:         var buyukLugat = ReadFile(@"Sources\RisaleNurSozlugu.txt");
828:         var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
829:         string text = AppResources.Installing + " " + AppResources.RisaleNur;
830:         //pbInstall.Value = 0;
831:         List<All> allNames = new List<All>();
832:         for (int i = 1; i < buyukLugatLines.Count; i++)
833:         {
834:             //pbInstall.Value = i;
835:             var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
836:             All allName = new All();
837:             allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
838:             allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? "" : information[1].TrimEnd().TrimStart();
839:             allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
840:             allName.AllSource = AppResources.RisaleNur;
841:             allName.AllNameSource = allName.AllName + " (" +
AppResources.RisaleNur + ")";
842:             allNames.Add(allName);
843:             //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
844:         }
845:         context.AllNames.InsertAllOnSubmit(allNames);
846:     }
847:
848:     if (cbEnglishVoll1.IsChecked == true)
849:     {
850:         var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.EnglishTurkishVoll1)).ToList() as List<All>;
851:         if (allList != null)
852:         {
853:             context.AllNames.DeleteAllOnSubmit(allList);
854:         }
855:         //burada metin belgeleri okunuyor ve veritabanındaki tablolara ekleme
yapiliyor
856:         var buyukLugat = ReadFile(@"Sources\IngilizceTurkceSozlukv1.txt");
857:         var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
858:         string text = AppResources.Installing + " " +
AppResources.EnglishTurkishVoll1;
859:         //pbInstall.Value = 0;
860:         List<All> allNames = new List<All>();
861:         for (int i = 1; i < buyukLugatLines.Count; i++)
862:         {
863:             //pbInstall.Value = i;
864:             var information = buyukLugatLines[i].Split('_').ToList() as

```



```

List<string>;
865:         All allName = new All();
866:         allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
867:         allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? "" : information[1].TrimEnd().TrimStart();
868:         allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
869:         allName.AllSource = AppResources.EnglishTurkishVol1;
870:         allName.AllNameSource = allName.AllName + " (" +
AppResources.EnglishTurkishVol1 + ")";
871:         allNames.Add(allName);
872:         //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
873:     }
874:     context.AllNames.InsertAllOnSubmit(allNames);
875: }
876:
877:     if (cbEnglishVol2.IsChecked == true)
878:     {
879:         var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.EnglishTurkishVol2)).ToList() as List<All>;
880:         if (allList != null)
881:         {
882:             context.AllNames.DeleteAllOnSubmit(allList);
883:         }
884:         //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
885:         var buyukLugat = ReadFile(@"Sources\IngilizceTurkceSozlukv2.txt");
886:         var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
887:         string text = AppResources.Installing + " " +
AppResources.EnglishTurkishVol2;
888:         //pbInstall.Value = 0;
889:         List<All> allNames = new List<All>();
890:         for (int i = 1; i < buyukLugatLines.Count; i++)
891:         {
892:             //pbInstall.Value = i;
893:             var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
894:             All allName = new All();
895:             allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
896:             allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? "" : information[1].TrimEnd().TrimStart();
897:             allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
898:             allName.AllSource = AppResources.EnglishTurkishVol2;
899:             allName.AllNameSource = allName.AllName + " (" +
AppResources.EnglishTurkishVol2 + ")";
900:             allNames.Add(allName);
901:             //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
902:         }
903:         context.AllNames.InsertAllOnSubmit(allNames);
904:     }
905:
906:     if (cbWordMeaning.IsChecked == true)
907:     {
908:         var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.WordMeaning)).ToList() as List<All>;
909:         if (allList != null)
910:         {
911:             context.AllNames.DeleteAllOnSubmit(allList);
912:         }
913:         //burada metin belgeleri okunuyor ve veritabanindaki tablolara ekleme
yapiliyor
914:         var buyukLugat = ReadFile(@"Sources\KelimeAnlamlari.txt");
915:         var buyukLugatLines = buyukLugat.Split('\n').ToList() as List<string>;
916:         string text = AppResources.Installing + " " +
AppResources.WordMeaning;
917:         //pbInstall.Value = 0;

```

```

918:         List<All> allNames = new List<All>();
919:         for (int i = 1; i < buyukLugatLines.Count; i++)
920:         {
921:             //pbInstall.Value = i;
922:             var information = buyukLugatLines[i].Split('_').ToList() as
List<string>;
923:             All allName = new All();
924:             allName.AllName = information[0].ToUpper().TrimEnd().TrimStart();
925:             allName.AllMeaning = information[1].TrimEnd().TrimStart() == null
? "" : information[1].TrimEnd().TrimStart();
926:             allName.AllNameMeaning = allName.AllName + " " +
allName.AllNameMeaning;
927:             allName.AllSource = AppResources.WordMeaning;
928:             allName.AllNameSource = allName.AllName + " (" +
AppResources.WordMeaning + ")";
929:             allNames.Add(allName);
930:             //txtInstalling.Text = text + " (" + i + "/" +
buyukLugatLines.Count + ")";
931:         }
932:         context.AllNames.InsertAllOnSubmit(allNames);
933:     }
934:     context.SubmitChanges();
935:     //pbInstall.Visibility = Visibility.Collapsed;
936:
937:     MessageBox.Show(AppResources.DictionariesInstalledSuccessfully);
938: }
939:
940: public string ReadFile(string filePath)
941: {
942:     var ResrouceStream = Application.GetResourceStream(new Uri(filePath,
UriKind.Relative));
943:     if (ResrouceStream != null)
944:     {
945:         Stream myFileStream = ResrouceStream.Stream;
946:         if (myFileStream.CanRead)
947:         {
948:             StreamReader myStreamReader = new StreamReader(myFileStream);
949:
950:             return myStreamReader.ReadToEnd();
951:         }
952:     }
953:     return "";
954: }
955:
956: private void btnUninstall_Click(object sender, RoutedEventArgs e)
957: {
958:     try
959:     {
960:
961:         if (cbBuyukLugat.IsChecked == false && cbComputer.IsChecked == false
&& cbEnglishVol1.IsChecked == false && cbEnglishVol2.IsChecked == false &&
cbOxford.IsChecked == false && cbRisaleNur.IsChecked == false && cbWordMeaning.IsChecked ==
false && cbGerman.IsChecked == false)
962:         {
963:             MessageBox.Show(AppResources.AtLeastOneDictionary);
964:         }
965:         else
966:         {
967:             using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
968:             {
969:                 SozlukKaldir(context);
970:             }
971:         }
972:     }
973:     catch (Exception ex)
974:     {
975:         MessageBox.Show(AppResources.SystemFault);
976:     }

```

```

977:         }
978:
979:         private void SozlukKaldir(AwesomeDictionaryDataContext context)
980:         {
981:             //pbInstall.Visibility = Visibility.Visible;
982:
983:             if (cbBuyukLugat.IsChecked == true)
984:             {
985:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.BuyukLugat)).ToList() as List<All>;
986:                 if (allList != null)
987:                 {
988:                     context.AllNames.DeleteAllOnSubmit(allList);
989:                 }
990:                 context.SubmitChanges();
991:             }
992:
993:             if (cbGerman.IsChecked == true)
994:             {
995:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.GermanTurkish)).ToList() as List<All>;
996:                 if (allList != null)
997:                 {
998:                     context.AllNames.DeleteAllOnSubmit(allList);
999:                 }
1000:                 context.SubmitChanges();
1001:             }
1002:
1003:             if (cbComputer.IsChecked == true)
1004:             {
1005:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.ComputerDictionary)).ToList() as List<All>;
1006:                 if (allList != null)
1007:                 {
1008:                     context.AllNames.DeleteAllOnSubmit(allList);
1009:                 }
1010:                 context.SubmitChanges();
1011:             }
1012:             if (cbOxford.IsChecked == true)
1013:             {
1014:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.OxfordDictionary)).ToList() as List<All>;
1015:                 if (allList != null)
1016:                 {
1017:                     context.AllNames.DeleteAllOnSubmit(allList);
1018:                 }
1019:                 context.SubmitChanges();
1020:             }
1021:             if (cbRisaleNur.IsChecked == true)
1022:             {
1023:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.RisaleNur)).ToList() as List<All>;
1024:                 if (allList != null)
1025:                 {
1026:                     context.AllNames.DeleteAllOnSubmit(allList);
1027:                 }
1028:                 context.SubmitChanges();
1029:             }
1030:             if (cbEnglishVol1.IsChecked == true)
1031:             {
1032:                 var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.EnglishTurkishVol1)).ToList() as List<All>;
1033:                 if (allList != null)
1034:                 {
1035:                     context.AllNames.DeleteAllOnSubmit(allList);
1036:                 }
1037:                 context.SubmitChanges();
1038:             }
1039:             if (cbEnglishVol2.IsChecked == true)

```

```

1040:         {
1041:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.EnglishTurkishVol2)).ToList() as List<All>;
1042:             if (allList != null)
1043:             {
1044:                 context.AllNames.DeleteAllOnSubmit(allList);
1045:             }
1046:             context.SubmitChanges();
1047:         }
1048:         if (cbWordMeaning.IsChecked == true)
1049:         {
1050:             var allList = context.AllNames.Where(j =>
j.AllSource.Equals(AppResources.WordMeaning)).ToList() as List<All>;
1051:             if (allList != null)
1052:             {
1053:                 context.AllNames.DeleteAllOnSubmit(allList);
1054:             }
1055:             context.SubmitChanges();
1056:         }
1057:         //context.SubmitChanges();
1058:
1059:         MessageBox.Show(AppResources.DictionariesUninstalledSuccessfully);
1060:     }
1061: }
1062: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.21 KelimeAnlamlari.cs

This is file KelimeAnlamlari.cs.

Body Source

```

1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class KelimeAnlamlari
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int KelimeAnlamlariId { get; set; }
19:
20:         [Column]
21:         public string KelimeAnlamlariName { get; set; }
22:
23:         [Column]
24:         public string KelimeAnlamlariMeaning { get; set; }
25:
26:         [Column]
27:         public string KelimeAnlamlariNameMeaning { get; set; }
28:

```

```
29:     }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.22 LanguageSettingsPage.xaml.cs

This is file LanguageSettingsPage.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Globalization;
4: using System.IO;
5: using System.Linq;
6: using System.Net;
7: using System.Threading;
8: using System.Windows;
9: using System.Windows.Controls;
10: using System.Windows.Media;
11: using System.Windows.Media.Imaging;
12: using System.Windows.Navigation;
13: using AwesomeDictionary.Resources;
14: using Microsoft.Phone.Controls;
15: using Microsoft.Phone.Shell;
16:
17: namespace AwesomeDictionary
18: {
19:     public partial class LanguageSettingsPage : PhoneApplicationPage
20:     {
21:         public LanguageSettingsPage()
22:         {
23:             InitializeComponent();
24:
25:             lstLanguage.Items.Clear();
26:             lstLanguage.Items.Add(AppResources.English);
27:             lstLanguage.Items.Add(AppResources.Turkish);
28:             lstLanguage.Items.Add(AppResources.German);
29:             //lstLanguage.Items.Add(AppResources.Spanish);
30:             //lstLanguage.Items.Add(AppResources.Russian);
31:             //lstLanguage.Items.Add(AppResources.Arabic);
32:             //lstLanguage.Items.Add(AppResources.Persian);
33:             //lstLanguage.Items.Add(AppResources.Chinese);
34:             //lstLanguage.Items.Add(AppResources.Italian);
35:             //lstLanguage.Items.Add(AppResources.French);
36:             //lstLanguage.Items.Add(AppResources.Japanese);
37:             //lstLanguage.Items.Add(AppResources.Sanskrit);
38:             //lstLanguage.Items.Add(AppResources.Thai);
39:
40:             lstLanguage.SelectedIndex = -1;
41:             lblLanguage.Text = AppResources.SelectLanguage;
42:             lblGeneralSettings.Text = AppResources.GeneralSettings;
43:
44:             SetBackgroundColor();
45:         }
46:
47:         protected override void OnNavigatedTo(NavigationEventArgs e)
48:         {
49:             base.OnNavigatedTo(e);
50:             SetBackgroundColor();
51:         }
52:     }
53: }
```

```

52:
53:     protected override void OnNavigatedFrom(NavigationEventArgs e)
54:     {
55:         base.OnNavigatedFrom(e);
56:     }
57:
58:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
59:     {
60:         if (this.NavigationService.CanGoBack)
61:         {
62:             this.NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
63:         }
64:     }
65:
66:     private void SetBackgroundColor()
67:     {
68:         AppSettings appSettings = new AppSettings();
69:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
70:         {
71:             appSettings = context.AppSettings.First() as AppSettings;
72:         }
73:
74:         if (appSettings.AppBackgroundImage != null)
75:         {
76:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
77:             BitmapImage image = new BitmapImage();
78:             image.SetSource(stream);
79:             ImageBrush ib = new ImageBrush();
80:             ib.ImageSource = image;
81:             this.LayoutRoot.Background = ib;
82:         }
83:         else
84:         {
85:             switch (appSettings.AppBackgroundColor)
86:             {
87:                 case "BLA":
88:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
89:                     break;
90:                 case "BLU":
91:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
92:                     break;
93:                 case "BRO":
94:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
95:                     break;
96:                 case "RED":
97:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
98:                     break;
99:                 case "GRE":
100:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
101:                     break;
102:                 case "GRA":
103:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
104:                     break;
105:                 case "YEL":
106:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
107:                     break;
108:                 case "ORA":
109:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
110:                     break;
111:                 case "PUR":
112:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
113:                     break;
114:                 default:

```

```
115:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
116:                 break;
117:             }
118:         }
119:     }
120:
121:     private void lstLanguage_SelectionChanged(object sender,
SelectionChangedEventArgs e)
122:     {
123:         int index = lstLanguage.SelectedIndex;
124:         string culture = "";
125:         string lang = "";
126:         if (index == 0)
127:         {
128:             culture = "en";
129:             lang = "EN";
130:         }
131:         else if (index == 1)
132:         {
133:             culture = "tr";
134:             lang = "TR";
135:         }
136:         else if (index == 2)
137:         {
138:             culture = "de";
139:             lang = "DE";
140:         }
141:         else if (index == 3)
142:         {
143:             culture = "ru";
144:             lang = "RU";
145:         }
146:         else if (index == 4)
147:         {
148:             culture = "ar";
149:             lang = "AR";
150:         }
151:         else if (index == 5)
152:         {
153:             culture = "fa-IR";
154:             lang = "FA";
155:         }
156:         else if (index == 6)
157:         {
158:             culture = "zh";
159:             lang = "ZH";
160:         }
161:         else if (index == 7)
162:         {
163:             culture = "it";
164:             lang = "IT";
165:         }
166:         else if (index == 8)
167:         {
168:             culture = "fr";
169:             lang = "FR";
170:         }
171:         else if (index == 9)
172:         {
173:             culture = "ja";
174:             lang = "JA";
175:         }
176:         else if (index == 10)
177:         {
178:             culture = "sa";
179:             lang = "SA";
180:         }
181:         else if (index == 11)
182:         {
```

```

183:         culture = "th";
184:         lang = "TH";
185:     }
186:     //else if (index == 3)
187:     //{
188:     //    culture = "es";
189:     //    lang = "ES";
190:     //}
191:     //else if (index == 4)
192:     //{
193:     //    culture = "ru";
194:     //    lang = "RU";
195:     //}
196:     //else if (index == 5)
197:     //{
198:     //    culture = "zh";
199:     //    lang = "AR";
200:     //}
201:     //else if (index == 6)
202:     //{
203:     //    culture = "ar";
204:     //    lang = "AR";
205:     //}
206:     //else if (index == 7)
207:     //{
208:     //    culture = "fa-IR";
209:     //    lang = "FA";
210:     //}
211:     //else if (index == 8)
212:     //{
213:     //    culture = "it";
214:     //    lang = "IT";
215:     //}
216:     //else if (index == 9)
217:     //{
218:     //    culture = "fr";
219:     //    lang = "FR";
220:     //}
221:     //else if (index == 10)
222:     //{
223:     //    culture = "pt";
224:     //    lang = "PT";
225:     //}
226:     else
227:     {
228:         culture = "en";
229:         lang = "EN";
230:     }
231:
232:     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
233:     {
234:         var appSettings = context.AppSettings;
235:         foreach (var appSetting in appSettings)
236:         {
237:             appSetting.AppLangName = lang;
238:         }
239:         context.SubmitChanges();
240:     }
241:
242:     CultureInfo newCulture = new CultureInfo(culture);
243:     Thread.CurrentThread.CurrentCulture = newCulture;
244:     Thread.CurrentThread.CurrentUICulture = newCulture;
245:     MessageBox.Show(AppResources.LanguageWarning);
246:     NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
247: }
248:
249: private void PhoneApplicationPage_Loaded(object sender, RoutedEventArgs e)

```



```
250:         {
251:             //SetBackgroundColor();
252:         }
253:     }
254: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.23 LocalizedStrings.cs

This is file LocalizedStrings.cs.

Body Source

```
1: ?using AwesomeDictionary.Resources;
2:
3: namespace AwesomeDictionary
4: {
5:     /// <summary>
6:     /// Provides access to string resources.
7:     /// </summary>
8:     public class LocalizedStrings
9:     {
10:         private static AppResources _localizedResources = new AppResources();
11:
12:         public AppResources LocalizedResources { get { return _localizedResources; } }
13:     }
14: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.24 MainPage.xaml.cs

This is file MainPage.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Text;
7: using System.Windows;
8: using System.Windows.Controls;
9: using System.Windows.Controls.Primitives;
10: using System.Windows.Input;
11: using System.Windows.Media;
12: using System.Windows.Media.Imaging;
13: using System.Windows.Navigation;
14: using Microsoft.Phone.Tasks;
15: using AwesomeDictionary.Resources;
16: using Microsoft.Phone.Controls;
17: using Microsoft.Phone.Shell;
18:
```

```

19: namespace AwesomeDictionary
20: {
21:     public partial class MainPage : PhoneApplicationPage
22:     {
23:         public MainPage()
24:         {
25:             InitializeComponent();
26:
27:             ApplicationBar = new ApplicationBar();
28:
29:             ApplicationBarIconButton button2 = new ApplicationBarIconButton();
30:             button2.IconUri = new Uri("/Assets/Search.png", UriKind.Relative);
31:             button2.Text = AppResources.Search;
32:             ApplicationBar.Buttons.Add(button2);
33:             button2.Click += new EventHandler(SearchButton_Click);
34:
35:             ApplicationBarIconButton button3 = new ApplicationBarIconButton();
36:             button3.IconUri = new Uri("/Assets/Settings.png", UriKind.Relative);
37:             button3.Text = AppResources.Settings;
38:             ApplicationBar.Buttons.Add(button3);
39:             button3.Click += new EventHandler(SettingsButton_Click);
40:
41:             //ApplicationBarIconButton button4 = new ApplicationBarIconButton();
42:             //button4.IconUri = new Uri("/Assets/Statistics.png", UriKind.Relative);
43:             //button4.Text = AppResources.Statistics;
44:             //ApplicationBar.Buttons.Add(button4);
45:             //button4.Click += new EventHandler(StatisticsButton_Click);
46:
47:             ApplicationBarMenuItem menuItem1 = new ApplicationBarMenuItem();
48:             menuItem1.Text = AppResources.About;
49:             ApplicationBar.MenuItems.Add(menuItem1);
50:             menuItem1.Click += new EventHandler(AboutMenuItem_Click);
51:
52:             SetBackgroundColor();
53:
54:             piFavourite.Header = AppResources.MyFavourites;
55:             piRandomWords.Header = AppResources.RandomWords;
56:
57:         }
58:
59:         private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
60:         {
61:             if (MessageBox.Show(AppResources.ExitAppQuestion,
AppResources.ExitApp, MessageBoxButton.OKCancel)
62:                 != MessageBoxResult.OK)
63:             {
64:                 e.Cancel = true;
65:             }
66:             else
67:             {
68:                 Application.Current.Terminate();
69:             }
70:         }
71:
72:         private void pNames_SelectionChanged(object sender, SelectionChangedEventArgs
e)
73:         {
74:             var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString);
75:             if (pNames.SelectedIndex == 0)
76:             {
77:                 List<Favourite> favourites = context.Favourites.OrderBy(j =>
j.FavouriteName).ToList() as List<Favourite>;
78:                 List<All> allNames = new List<All>();
79:                 for (int i = 0; i < favourites.Count; i++)
80:                 {
81:                     var allName = context.AllNames.Where(j =>
j.AllId.Equals(favourites[i].FavouriteAllId)).SingleOrDefault() as All;

```

```

83:         allNames.Add(allName);
84:     }
85:     List<AlphaKeyGroup<All>> DataSource =
AlphaKeyGroup<All>.CreateGroups(allNames,
86:         System.Threading.Thread.CurrentThread.CurrentUICulture,
87:         (All a) => { return a.AllName; }, true);
88:     llsFavourites.ItemsSource = DataSource;
89: }
90:
91:     else if (pNames.SelectedIndex == 1)
92:     {
93:         List<All> allNames = new List<All>();
94:         if (context.AllNames.ToList().Count > 0)
95:         {
96:             List<int> sayiListesi = new List<int>();
97:             Random random = new Random();
98:             int tableCount = context.AllNames.ToList().Count;
99:             for (int i = 0; i < 10; i++)
100:             {
101:                 int sayi = random.Next(i, tableCount);
102:                 sayiListesi.Add(sayi);
103:             }
104:             for (int k = 0; k < sayiListesi.Count(); k++)
105:             {
106:                 //int sayi = random.Next(0, 500000);
107:                 var allName = context.AllNames.Where(j =>
j.AllId.Equals(sayiListesi[k])).SingleOrDefault() as All;
108:                 //var allName = context.AllNames.Where(j =>
j.AllId.Equals(sayi)).SingleOrDefault() as All;
109:                 if (allName != null)
110:                 {
111:                     allNames.Add(allName);
112:                 }
113:                 //if(allNames.Count == 10)
114:                 //{
115:                     break;
116:                 //}
117:             }
118:         }
119:         List<AlphaKeyGroup<All>> DataSource =
AlphaKeyGroup<All>.CreateGroups(allNames,
120:             System.Threading.Thread.CurrentThread.CurrentUICulture,
121:             (All a) => { return a.AllName; }, true);
122:         llsRandomWords.ItemsSource = DataSource;
123:     }
124: }
125:
126:     private void SettingsButton_Click(object sender, EventArgs e)
127:     {
128:         NavigationService.Navigate(new Uri("/GeneralSettingsPage.xaml",
UriKind.Relative));
129:     }
130:
131:     private void SearchButton_Click(object sender, EventArgs e)
132:     {
133:         NavigationService.Navigate(new Uri("/SearchPage.xaml", UriKind.Relative));
134:     }
135:
136:     private void AboutMenuItem_Click(object sender, EventArgs e)
137:     {
138:         NavigationService.Navigate(new Uri("/AboutPage.xaml", UriKind.Relative));
139:     }
140:     private void StatisticsButton_Click(object sender, EventArgs e)
141:     {
142:         //NavigationService.Navigate(new Uri("/StatisticsPage.xaml",
UriKind.Relative));
143:     }
144:
145:     private void AddNameButton_Click(object sender, EventArgs e)

```

```

146:         {
147:             NavigationService.Navigate(new Uri("/NameDetailPage.xaml",
UriKind.Relative));
148:             //PhoneApplicationPage_Loaded(this, new RoutedEventArgs());
149:         }
150:
151:
152:         private void llsFavourites_SelectionChanged(object sender,
SelectionChangedEventArgs e)
153:         {
154:             //var favourite = llsFavourites.SelectedItem as Favourite;
155:             var favourite = llsFavourites.SelectedItem as All;
156:             NavigationService.Navigate(new Uri("/NameDetailPage.xaml#" +
favourite.AllId, UriKind.Relative));
157:         }
158:
159:         private void SetBackgroundColor()
160:         {
161:             AppSettings appSettings = new AppSettings();
162:             using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
163:             {
164:                 appSettings = context.AppSettings.First() as AppSettings;
165:             }
166:
167:             if (appSettings.AppBackgroundImage != null)
168:             {
169:                 MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
170:                 BitmapImage image = new BitmapImage();
171:                 image.SetSource(stream);
172:                 ImageBrush ib = new ImageBrush();
173:                 ib.ImageSource = image;
174:                 this.LayoutRoot.Background = ib;
175:             }
176:             else
177:             {
178:                 switch (appSettings.AppBackgroundColor)
179:                 {
180:                     case "BLA":
181:                         this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
182:                         break;
183:                     case "BLU":
184:                         this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
185:                         break;
186:                     case "BRO":
187:                         this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
188:                         break;
189:                     case "RED":
190:                         this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
191:                         break;
192:                     case "GRE":
193:                         this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
194:                         break;
195:                     case "GRA":
196:                         this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
197:                         break;
198:                     case "YEL":
199:                         this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
200:                         break;
201:                     case "ORA":
202:                         this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
203:                         break;
204:                     case "PUR":
205:                         this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
206:                         break;
207:                     default:

```

```
208:                                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
209:                                     break;
210:                                 }
211:                             }
212:                         }
213:
214:         private void llsRandomWords_SelectionChanged(object sender,
SelectionChangedEventArgs e)
215:         {
216:             var randomWord = llsRandomWords.SelectedItem as All;
217:             NavigationService.Navigate(new Uri("/NameDetailPage.xaml#" +
randomWord.AllId, UriKind.Relative));
218:         }
219:     }
220: }
221: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.25 NameDetailPage.xaml.cs

This is file NameDetailPage.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Text;
7: using System.Windows;
8: using System.Windows.Controls;
9: using System.Windows.Controls.Primitives;
10: using System.Windows.Input;
11: using System.Windows.Media;
12: using System.Windows.Media.Imaging;
13: using System.Windows.Navigation;
14: using Microsoft.Phone.Tasks;
15: using AwesomeDictionary.Resources;
16: using Microsoft.Phone.Controls;
17: using Microsoft.Phone.Shell;
18:
19: namespace AwesomeDictionary
20: {
21:     public partial class NameDetailPage : PhoneApplicationPage
22:     {
23:         public string pageName;
24:         public int wordId;
25:         double InputHeight = 0.0;
26:         public bool flag;
27:         public bool isFilled;
28:         public double ratingValue = 0;
29:         string gender;
30:         All allName = new All();
31:
32:         public NameDetailPage()
33:         {
34:             InitializeComponent();
35:
36:             ApplicationBar = new ApplicationBar();
37:         }
38:     }
39: }
```

```

38:         //ApplicationBarIconButton button1 = new ApplicationBarIconButton();
39:         //button1.IconUri = new Uri("/Assets/Save.png", UriKind.Relative);
40:         //button1.Text = "Kaydet";
41:         //ApplicationBar.Buttons.Add(button1);
42:         //button1.Click += new EventHandler(SaveButton_Click);
43:
44:         ApplicationBarIconButton button2 = new ApplicationBarIconButton();
45:         button2.IconUri = new Uri("/Assets/SendWithMail.png", UriKind.Relative);
46:         button2.Text = AppResources.SendWithEmail;
47:         ApplicationBar.Buttons.Add(button2);
48:         button2.Click += new EventHandler(SendMailButton_Click);
49:
50:         ApplicationBarIconButton button3 = new ApplicationBarIconButton();
51:         button3.IconUri = new Uri("/Assets/SendWithSMS.png", UriKind.Relative);
52:         button3.Text = AppResources.SendWithSMS;
53:         ApplicationBar.Buttons.Add(button3);
54:         button3.Click += new EventHandler(SendSMSButton_Click);
55:
56:         ApplicationBarIconButton button4 = new ApplicationBarIconButton();
57:         button4.IconUri = new Uri("/Assets/Share.png", UriKind.Relative);
58:         button4.Text = AppResources.Share;
59:         ApplicationBar.Buttons.Add(button4);
60:         button4.Click += new EventHandler(ShareNameButton_Click);
61:
62:         isFilled = false;
63:
64:         //ApplicationBarMenuItem menuItem1 = new ApplicationBarMenuItem();
65:         //menuItem1.Text = "Sil";
66:         //ApplicationBar.MenuItems.Add(menuItem1);
67:         //menuItem1.Click += new EventHandler(DeleteNameMenuItem_Click);
68:
69:         //List<string> genderList = new List<string>();
70:         //genderList.Add("Lütfen seçiniz");
71:         //genderList.Add("Erkek");
72:         //genderList.Add("Kadin");
73:         //genderList.Add("Erkek-Kadin");
74:         //lpGender.ItemsSource = genderList;
75:         //lpGender.SelectedIndex = 0;
76:
77:         SetBackgroundColor();
78:     }
79:
80:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
81:     {
82:         if (pageName.Contains("/SearchPage.xaml"))
83:         {
84:             //this.NavigationService.Navigate(new Uri("/SearchPage.xaml",
UriKind.Relative));
85:         }
86:         else
87:         {
88:             this.NavigationService.Navigate(new Uri("/MainPage.xaml",
UriKind.Relative));
89:         }
90:     }
91:
92:     protected override void OnNavigatedTo(NavigationEventArgs e)
93:     {
94:         base.OnNavigatedTo(e);
95:
96:         var lastPage = NavigationService.BackStack.FirstOrDefault();
97:         pageName = lastPage.Source.ToString();
98:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
99:         {
100:             var appSettings = context.AppSettings.First();
101:
102:             FontFamily temp = new FontFamily(appSettings.FontFamily);

```

```

103:         double fontsize = double.Parse(appSettings.FontSize);
104:         txtMeaning.FontFamily = temp;
105:         txtMeaning.FontSize = fontsize;
106:     }
107:
108:     txtMeaning.IsEnabled = false;
109:     isFilled = false;
110:     //SetBackgroundColor();
111:     //while (NavigationService.CanGoBack)
112:     //NavigationService.RemoveBackEntry();
113:
114: }
115:
116: protected override void OnNavigatedFrom(NavigationEventArgs e)
117: {
118:     base.OnNavigatedFrom(e);
119:     //while (NavigationService.CanGoBack)
120:     //NavigationService.RemoveBackEntry();
121:
122: }
123:
124: protected override void OnFragmentNavigation(FragmentNavigationEventArgs e)
125: {
126:     // displays "Fragment: Detail"
127:     //MessageBox.Show("Folder Id: " + e.Fragment);
128:     base.OnFragmentNavigation(e);
129:     //string fragmentName = e.Fragment.ToString();
130:     wordId = Convert.ToInt32(e.Fragment);
131:     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
132:     {
133:         var name = context.AllNames.Where(j =>
j.AllId.Equals(wordId)).SingleOrDefault() as All;
134:         allName = name;
135:         //for (int i = 0; i < lpGender.Items.Count; i++)
136:         //{
137:             // if (lpGender.Items[i].ToString() == name.AllGender)
138:             // {
139:                 // lpGender.SelectedIndex = i;
140:                 // break;
141:             // }
142:         //}
143:         lblNameDetail.Text = name.AllName;
144:         txtMeaning.Text = name.AllMeaning + Environment.NewLine +
Environment.NewLine + AppResources.Source + ":" + name.AllSource + "";
145:         txtMeaning.IsEnabled = false;
146:         //txtMeaning.Text = name.AllMeaning;
147:
148:         var favourite = context.Favourites.Where(j =>
j.FavouriteAllId.Equals(wordId)).SingleOrDefault() as Favourite;
149:         if (favourite != null)
150:         {
151:             ApplicationBarItem menuItem4 = new ApplicationBarItem();
152:             menuItem4.Text = AppResources.RemoveFromFavourites;
153:             ApplicationBar.MenuItems.Add(menuItem4);
154:             menuItem4.Click += new
EventHandler(RemoveFavouritesMenuItem_Click);
155:         }
156:         else
157:         {
158:             ApplicationBarItem menuItem3 = new ApplicationBarItem();
159:             menuItem3.Text = AppResources.AddToFavourites;
160:             ApplicationBar.MenuItems.Add(menuItem3);
161:             menuItem3.Click += new EventHandler(AddFavouritesMenuItem_Click);
162:         }
163:
164:         //var myUpdate = context.MyUpdates.Where(j =>
j.MyUpdateName.Equals(fragmentName)).SingleOrDefault() as MyUpdate; ;
165:         //if (myUpdate != null)

```

```

166:         //{
167:         //    ApplicationBarItem menuItem2 = new ApplicationBarItem();
168:         //    menuItem2.Text = "Sisteme Eklenmesi İçin Gönder";
169:         //    ApplicationBar.MenuItems.Add(menuItem2);
170:         //    menuItem2.Click += new EventHandler(SaveAndSendMenuItem_Click);
171:         //}
172:
173:
174:
175:         isFilled = true;
176:         //pvName.SelectedIndex = 0;
177:     }
178: }
179:
180: private void SetBackgroundColor()
181: {
182:     AppSettings appSettings = new AppSettings();
183:     using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
184:     {
185:         appSettings = context.AppSettings.First() as AppSettings;
186:     }
187:
188:     if (appSettings.AppBackgroundImage != null)
189:     {
190:         MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
191:         BitmapImage image = new BitmapImage();
192:         image.SetSource(stream);
193:         ImageBrush ib = new ImageBrush();
194:         ib.ImageSource = image;
195:         this.LayoutRoot.Background = ib;
196:     }
197:     else
198:     {
199:         switch (appSettings.AppBackgroundColor)
200:         {
201:             case "BLA":
202:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
203:                 break;
204:             case "BLU":
205:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
206:                 break;
207:             case "BRO":
208:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
209:                 break;
210:             case "RED":
211:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
212:                 break;
213:             case "GRE":
214:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
215:                 break;
216:             case "GRA":
217:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
218:                 break;
219:             case "YEL":
220:                 this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
221:                 break;
222:             case "ORA":
223:                 this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
224:                 break;
225:             case "PUR":
226:                 this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
227:                 break;
228:             default:
229:                 this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
230:                 break;

```



```

231:         }
232:     }
233: }
234:
235: private void txtName_KeyDown(object sender, KeyEventArgs e)
236: {
237:     //if (e.Key == Key.Enter)
238:     //{
239:         pvName.SelectedIndex = 2;
240:         txtMeaning.Focus();
241:     //}
242: }
243:
244: private void txtMeaning_TextChanged(object sender, TextChangedEventArgs e)
245: {
246:     Dispatcher.BeginInvoke(() =>
247:     {
248:         double CurrentInputHeight = txtMeaning.ActualHeight;
249:
250:         if (CurrentInputHeight > InputHeight)
251:         {
252:             svMeaning.ScrollToVerticalOffset(svMeaning.VerticalOffset +
CurrentInputHeight - InputHeight);
253:         }
254:
255:         InputHeight = CurrentInputHeight;
256:     });
257: }
258:
259: private void txtMeaning_GotFocus(object sender, RoutedEventArgs e)
260: {
261:     App.RootFrame.RenderTransform = new CompositeTransform();
262:     flag = true;
263: }
264:
265: private void txtMeaning_Tap(object sender,
System.Windows.Input.GestureEventArgs e)
266: {
267:     txtMeaning.Focus();
268:     //txtMeaning.Select(txtMeaning.Text.Length, 1);
269:     svMeaning.ScrollToVerticalOffset(e.GetPosition(txtMeaning).Y - 80);
270: }
271:
272: private void txtMeaning_LostFocus(object sender, RoutedEventArgs e)
273: {
274:     if (!flag) return;
275:     txtMeaning.Focus();
276:     flag = false;
277:     this.pnlKeyboardPlaceholder.Visibility = Visibility.Collapsed;
278: }
279:
280: private void txtMeaning_KeyDown(object sender, KeyEventArgs e)
281: {
282:     if (e.Key == Key.Enter)
283:     {
284:         svMeaning.ScrollToVerticalOffset(txtMeaning.ActualHeight);
285:     }
286: }
287:
288: private void svMeaning_GotFocus(object sender, RoutedEventArgs e)
289: {
290:     this.svMeaning.ScrollToVerticalOffset(this.txtMeaning.ActualHeight);
291:     this.svMeaning.UpdateLayout();
292: }
293:
294: private void SendSMSButton_Click(object sender, EventArgs e)
295: {
296:     SmsComposeTask smsComposeTask = new SmsComposeTask();
297:

```

```

298:         smsComposeTask.To = "";
299:         smsComposeTask.Body = CreateSendMaterial();
300:
301:         smsComposeTask.Show();
302:         //MessageBox.Show(AppResources.SuccessfulSendWithSMS);
303:     }
304:
305:     private void ShareNameButton_Click(object sender, EventArgs e)
306:     {
307:         ShareStatusTask shareStatusTask = new ShareStatusTask();
308:
309:         shareStatusTask.Status = CreateSendMaterial();
310:
311:         shareStatusTask.Show();
312:         //MessageBox.Show(AppResources.SuccessfulSendWithSMS);
313:     }
314:
315:     private void SendMailButton_Click(object sender, EventArgs e)
316:     {
317:         // burada birden fazla e-posta hesabi varsa birini seçmesi söyleniyor
318:         //EmailAddressChooserTask emailAddressChooserTask;
319:         //emailAddressChooserTask = new EmailAddressChooserTask();
320:         //emailAddressChooserTask.Completed += new
321:         //EventHandler<EmailResult>(emailAddressChooserTask_Completed);
322:         //emailAddressChooserTask.Show();
323:
324:         EmailComposeTask emailComposeTask = new EmailComposeTask();
325:
326:         emailComposeTask.Subject = AppResources.WordAndMeaning + " (" +
327:         lblNameDetail.Text + ")";
328:         emailComposeTask.Body = CreateSendMaterial();
329:         emailComposeTask.To = "";
330:         emailComposeTask.Cc = "";
331:         emailComposeTask.Bcc = "";
332:
333:         emailComposeTask.Show();
334:         //MessageBox.Show(AppResources.SuccessfulSendWithMail);
335:     }
336:
337:     private string CreateSendMaterial()
338:     {
339:         StringBuilder sb = new StringBuilder();
340:         sb.AppendLine(AppResources.Word + ": " + lblNameDetail.Text);
341:         sb.AppendLine(AppResources.Meaning + ": " + txtMeaning.Text);
342:         //sb.AppendLine("Cinsiyeti: " + lpGender.Items[lpGender.SelectedIndex]);
343:         sb.AppendLine();
344:         sb.AppendLine(AppResources.SendWithAwesomeDictionaryApp);
345:         return sb.ToString();
346:     }
347:
348:     private void AddFavouritesMenuItem_Click(object sender, EventArgs e)
349:     {
350:         using (var context = new
351:         AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
352:         {
353:             var favourites = context.Favourites.Where(j =>
354:             j.FavouriteName.Equals(lblNameDetail.Text)).ToList() as List<Favourite>;
355:             if (favourites.Count > 0)
356:             {
357:                 MessageBox.Show(AppResources.WordAlreadyFavourite);
358:             }
359:             else
360:             {
361:                 Favourite favourite = new Favourite();
362:                 favourite.FavouriteName = lblNameDetail.Text;
363:                 favourite.FavouriteAllId = wordId;
364:                 context.Favourites.InsertOnSubmit(favourite);
365:                 context.SubmitChanges();
366:             }
367:         }
368:     }

```

```

363:         MessageBox.Show(AppResources.WordAddedFavouriteSuccessfully);
364:     }
365: }
366:
367:
368: private void RemoveFavouritesMenuItem_Click(object sender, EventArgs e)
369: {
370:     if (MessageBox.Show(AppResources.RemoveFromFavouriteQuestion,
371:         AppResources.RemoveFromFavourite, MessageBoxButtons.OKCancel)
372:         != DialogResult.OK)
373:     {
374:     }
375:     else
376:     {
377:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
378:         {
379:             var favourite = context.Favourites.Where(j =>
j.FavouriteName.Equals(lblNameDetail.Text)).ToList() as List<Favourite>;
380:             context.Favourites.DeleteAllOnSubmit(favourite);
381:             context.SubmitChanges();
382:             MessageBox.Show(AppResources.WordRemovedFavouriteSuccessfully);
383:         }
384:     }
385: }
386:
387: private void SaveAndSendMenuItem_Click(object sender, EventArgs e)
388: {
389:     //EmailComposeTask emailComposeTask = new EmailComposeTask();
390:
391:     //emailComposeTask.Subject = "Isim ve Anlami (" + txtName.Text + ")";
392:     //emailComposeTask.Body = CreateSendMaterial();
393:     //emailComposeTask.To = "coderserdar@outlook.com";
394:     //emailComposeTask.Cc = "";
395:     //emailComposeTask.Bcc = "";
396:
397:     //emailComposeTask.Show();
398: }
399:
400: private void DeleteNameMenuItem_Click(object sender, EventArgs e)
401: {
402:     //if (isFilled == true)
403:     //{
404:         // if (MessageBox.Show("Isim Silmek Istediginize Emin Misiniz?",
405:         //     "Isim Sil", MessageBoxButtons.OKCancel)
406:         //     != DialogResult.OK)
407:         // {
408:         // }
409:         // else
410:         // {
411:             // using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
412:             // {
413:
414:                 // switch (allName.AllGender)
415:                 // {
416:                     // case "Erkek":
417:                         // var maleName = context.MaleNames.Where(j =>
j.MaleName.Equals(allName.AllName)).SingleOrDefault() as Male;
418:                         // context.MaleNames.DeleteOnSubmit(maleName);
419:                         // break;
420:                     // case "Kadin":
421:                         // var femaleName = context.FemaleNames.Where(j =>
j.FemaleName.Equals(allName.AllName)).SingleOrDefault() as Female;
422:                         // context.FemaleNames.DeleteOnSubmit(femaleName);
423:                         // break;
424:                     // case "Erkek-Kadin":
425:                         // var unisexName = context.UnisexNames.Where(j =>
j.UnisexName.Equals(allName.AllName)).SingleOrDefault() as Unisex;

```

```

426:         //             context.UnisexNames.DeleteOnSubmit(unisexName);
427:         //             break;
428:         //         default:
429:         //             break;
430:         //     }
431:
432:         //         var allNames = context.AllNames.Where(j =>
433: j.AllName.Equals(allName.AllName)).SingleOrDefault() as All;
434:         //         context.AllNames.DeleteOnSubmit(allNames);
435:
436:         //         var myUpdates = context.MyUpdates.Where(j =>
437: j.MyUpdateName.Equals(allName.AllName)).SingleOrDefault() as MyUpdate;
438:         //         if (myUpdates != null)
439:         //         {
440:         //             context.MyUpdates.DeleteOnSubmit(myUpdates);
441:         //         }
442:
443:         //         var favourites = context.Favourites.Where(j =>
444: j.FavouriteName.Equals(allName.AllName)).SingleOrDefault() as Favourite;
445:         //         if (favourites != null)
446:         //         {
447:         //             context.Favourites.DeleteOnSubmit(favourites);
448:         //         }
449:         //     }
450:         // }
451:         // }
452:         //this.NavigationService.Navigate(new Uri("/MainPage.xaml",
453: UriKind.Relative));
454:     }
455:     private void SaveButton_Click(object sender, EventArgs e)
456:     {
457:         //using (var context = new
458: AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
459:         // {
460:         //     if (isFilled == false)
461:         //     {
462:         //         if (lpGender.SelectedIndex < 1 ||
463: txtName.Text.TrimEnd().TrimStart().Length < 1 ||
464: txtMeaning.Text.TrimEnd().TrimStart().Length < 1)
465:         //         {
466:         //             MessageBox.Show("Bütün Alanlari Dolduraniz Gerekmetedir.");
467:         //         }
468:         //         else
469:         //         {
470:         //             var allName = context.AllNames.Where(j =>
471: j.AllName.ToLower().Equals(txtName.Text.TrimEnd().TrimStart().ToLower())).SingleOrDefault()
472: as All;
473:         //             if (allName != null)
474:         //             {
475:         //                 MessageBox.Show("Bu Isim Sistemde Mevcut");
476:         //             }
477:         //             else
478:         //             {
479:         //                 switch (lpGender.SelectedIndex)
480:         //                 {
481:         //                     case 1:
482:         //                         Male maleName = new Male();
483:         //                         maleName.MaleName = txtName.Text.ToUpper();
484:         //                         maleName.MaleMeaning = txtMeaning.Text;
485:         //                         context.MaleNames.InsertOnSubmit(maleName);
486:         //                         break;
487:         //                     case 2:
488:         //                         Female femaleName = new Female();
489:         //                         femaleName.FemaleName = txtName.Text.ToUpper();

```

```

486:         //         femaleName.FemaleMeaning = txtMeaning.Text;
487:         //         context.FemaleNames.InsertOnSubmit(femaleName);
488:         //         break;
489:         //         case 3:
490:         //             Unisex unisexName = new Unisex();
491:         //             unisexName.UnisexName = txtName.Text.ToUpper();
492:         //             unisexName.UnisexMeaning = txtMeaning.Text;
493:         //             context.UnisexNames.InsertOnSubmit(unisexName);
494:         //             break;
495:         //         default:
496:         //             break;
497:         //     }
498:
499:         //     All allNameTemp = new All();
500:         //     allNameTemp.AllName = txtName.Text.ToUpper();
501:         //     allNameTemp.AllMeaning = txtMeaning.Text;
502:         //     allNameTemp.AllNameMeaning = txtName.Text + " " +
txtMeaning.Text;
503:         //     allNameTemp.AllGender =
lpGender.Items[lpGender.SelectedIndex].ToString();
504:         //     context.AllNames.InsertOnSubmit(allNameTemp);
505:
506:         //     MyUpdate myUpdate = new MyUpdate();
507:         //     myUpdate.MyUpdateName = txtName.Text.ToUpper();
508:         //     context.MyUpdates.InsertOnSubmit(myUpdate);
509:
510:         //     context.SubmitChanges();
511:         //     MessageBox.Show("Isim, Sisteme Basarili Bir Sekilde
Eklendi");
512:
513:         //     }
514:         //     }
515:         //     }
516:         //     else
517:         //     {
518:         //         if (allName.AllGender ==
lpGender.Items[lpGender.SelectedIndex].ToString() && allName.AllMeaning ==
txtMeaning.Text.TrimEnd().TrimStart())
519:         //         {
520:         //             MessageBox.Show("Herhangi bir degisiklik yapmadiniz.");
521:         //         }
522:         //         else
523:         //         {
524:         //             // önce eski kayıt siliniyor
525:         //             switch (allName.AllGender)
526:         //             {
527:         //                 case "Erkek":
528:         //                     var maleName = context.MaleNames.Where(j =>
j.MaleName.Equals(allName.AllName)).SingleOrDefault() as Male;
529:         //                     context.MaleNames.DeleteOnSubmit(maleName);
530:         //                     break;
531:         //                 case "Kadin":
532:         //                     var femaleName = context.FemaleNames.Where(j =>
j.FemaleName.Equals(allName.AllName)).SingleOrDefault() as Female;
533:         //                     context.FemaleNames.DeleteOnSubmit(femaleName);
534:         //                     break;
535:         //                 case "Erkek-Kadin":
536:         //                     var unisexName = context.UnisexNames.Where(j =>
j.UnisexName.Equals(allName.AllName)).SingleOrDefault() as Unisex;
537:         //                     context.UnisexNames.DeleteOnSubmit(unisexName);
538:         //                     break;
539:         //                 default:
540:         //                     break;
541:         //             }
542:
543:         //             // sonrasinda yeni kayıt ekleniyor
544:         //             switch (lpGender.SelectedIndex)
545:         //             {
546:         //                 case 1:

```

```

547:         //         Male maleName = new Male();
548:         //         maleName.MaleName = txtName.Text.ToUpper();
549:         //         maleName.MaleMeaning = txtMeaning.Text;
550:         //         context.MaleNames.InsertOnSubmit(maleName);
551:         //         break;
552:         //     case 2:
553:         //         Female femaleName = new Female();
554:         //         femaleName.FemaleName = txtName.Text.ToUpper();
555:         //         femaleName.FemaleMeaning = txtMeaning.Text;
556:         //         context.FemaleNames.InsertOnSubmit(femaleName);
557:         //         break;
558:         //     case 3:
559:         //         Unisex unisexName = new Unisex();
560:         //         unisexName.UnisexName = txtName.Text.ToUpper();
561:         //         unisexName.UnisexMeaning = txtMeaning.Text;
562:         //         context.UnisexNames.InsertOnSubmit(unisexName);
563:         //         break;
564:         //     default:
565:         //         break;
566:         //     }
567:
568:         //         var all = context.AllNames.Where(j =>
569:         j.AllName.Equals(allName.AllName)).Select(j => j);
570:         //         foreach (var item in all)
571:         //         {
572:         //             item.AllGender =
573:         lpGender.Items[lpGender.SelectedIndex].ToString();
574:         //             item.AllMeaning = txtMeaning.Text.TrimEnd().TrimStart();
575:         //             item.AllNameMeaning = item.AllName + " " +
576:         item.AllMeaning;
577:         //         }
578:         //         var myUpdateTemp = context.MyUpdates.Where(j =>
579:         j.MyUpdateName.Equals(allName.AllName)).SingleOrDefault() as MyUpdate;
580:         //         if (myUpdateTemp == null)
581:         //         {
582:         //             MyUpdate myUpdate = new MyUpdate();
583:         //             myUpdate.MyUpdateName = allName.AllName;
584:         //             context.MyUpdates.InsertOnSubmit(myUpdate);
585:         //         }
586:         //         context.SubmitChanges();
587:         //         MessageBox.Show("Isim Bilgisi Basarili Bir Sekilde
588:         Güncellendi");
589:         //     }
590:     }
591:
592:     private void lpGender_SelectionChanged(object sender,
593:     SelectionChangedEventArgs e)
594:     {
595:         //if (lpGender.SelectedIndex != 0)
596:         //{
597:         //     gender = lpGender.Items[lpGender.SelectedIndex].ToString();
598:         //     pvName.SelectedIndex = 1;
599:         //     txtName.Focus();
600:         // }
601:
602:     private void PhoneApplicationPage_Loaded(object sender, RoutedEventArgs e)
603:     {
604:
605:     }
606:
607:     }
608: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.26 OxfordEnglishEnglish.cs

This is file OxfordEnglishEnglish.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class OxfordEnglishEnglish
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int OxfordId { get; set; }
19:
20:         [Column]
21:         public string OxfordName { get; set; }
22:
23:         [Column]
24:         public string OxfordMeaning { get; set; }
25:
26:         [Column]
27:         public string OxfordNameMeaning { get; set; }
28:
29:     }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.27 RisaleNur.cs

This is file RisaleNur.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.Data.Linq;
4: using System.Data.Linq.Mapping;
5: using System.Linq;
6: using System.Text;
7: using System.Threading.Tasks;
```

```
8:
9: namespace AwesomeDictionary
10: {
11:     [Table]
12:     public class RisaleNur
13:     {
14:         [Column(IsPrimaryKey = true,
15:             IsDbGenerated = true,
16:             DbType = "INT NOT NULL Identity",
17:             CanBeNull = false)]
18:         public int RisaleNurId { get; set; }
19:
20:         [Column]
21:         public string RisaleNurName { get; set; }
22:
23:         [Column]
24:         public string RisaleNurMeaning { get; set; }
25:
26:         [Column]
27:         public string RisaleNurNameMeaning { get; set; }
28:     }
29: }
30: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.28 SearchPage.xaml.cs

This is file SearchPage.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Text;
7: using System.Windows;
8: using System.Windows.Controls;
9: using System.Windows.Controls.Primitives;
10: using System.Windows.Input;
11: using System.Windows.Media;
12: using System.Windows.Media.Imaging;
13: using System.Windows.Navigation;
14: using Microsoft.Phone.Tasks;
15: using AwesomeDictionary.Resources;
16: using Microsoft.Phone.Controls;
17: using Microsoft.Phone.Shell;
18:
19: namespace AwesomeDictionary
20: {
21:     public partial class SearchPage : PhoneApplicationPage
22:     {
23:         public SearchPage()
24:         {
25:             InitializeComponent();
26:             SetBackgroundColor();
27:
28:             txtSearchResult.Text = AppResources.SearchResults;
29:             txtSearchWithMeaning.Text = AppResources.SearchInMeanings;
30:             lblSearch.Text = AppResources.Search;
```



```

31:         //btnSearch.Content = AppResources.Search;
32:         //lstSearch.SelectedIndex = -1;
33:     }
34:
35:     private void SetBackgroundColor()
36:     {
37:         AppSettings appSettings = new AppSettings();
38:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
39:         {
40:             appSettings = context.AppSettings.First() as AppSettings;
41:         }
42:
43:         if (appSettings.AppBackgroundImage != null)
44:         {
45:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
46:             BitmapImage image = new BitmapImage();
47:             image.SetSource(stream);
48:             ImageBrush ib = new ImageBrush();
49:             ib.ImageSource = image;
50:             this.LayoutRoot.Background = ib;
51:         }
52:         else
53:         {
54:             switch (appSettings.AppBackgroundColor)
55:             {
56:                 case "BLA":
57:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
58:                     break;
59:                 case "BLU":
60:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
61:                     break;
62:                 case "BRO":
63:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
64:                     break;
65:                 case "RED":
66:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
67:                     break;
68:                 case "GRE":
69:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
70:                     break;
71:                 case "GRA":
72:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
73:                     break;
74:                 case "YEL":
75:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
76:                     break;
77:                 case "ORA":
78:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
79:                     break;
80:                 case "PUR":
81:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
82:                     break;
83:                 default:
84:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
85:                     break;
86:             }
87:         }
88:     }
89:
90:     private void btnSearch_Click(object sender, RoutedEventArgs e)
91:     {
92:         lstSearch.Items.Clear();
93:         var nameList = new List<All>();
94:         if (txtSearch.Text.TrimStart().TrimEnd().Length < 1)
95:         {

```

```

96:         MessageBox.Show(AppResources.SearchTrimFault);
97:     }
98:     else
99:     {
100:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
101:         {
102:             if (cbSearchWithMeaning.IsChecked == true)
103:             {
104:                 nameList = context.AllNames.Where(j =>
j.AllNameMeaning.ToLower().Contains(txtSearch.Text.TrimEnd().TrimStart().ToLower())).ToList()
105:             }
106:             else
107:             {
108:                 nameList = context.AllNames.Where(j =>
j.AllName.ToLower().Contains(txtSearch.Text.TrimEnd().TrimStart().ToLower())).ToList() as
List<All>;
109:             }
110:             //var noteList = context.Notes.ToList() as List<Note>;
111:
112:             if (nameList != null)
113:             {
114:                 txtSearchResult.Text = AppResources.SearchResults + " (" +
nameList.Count() + ")";
115:             }
116:
117:             //List<All> allNames = nameList.OrderBy(j => j.AllName).ToList()
as List<All>;
118:             //List<AlphaKeyGroup<All>> DataSource =
AlphaKeyGroup<All>.CreateGroups(allNames,
119:             //    System.Threading.Thread.CurrentThread.CurrentUICulture,
120:             //    (All a) => { return a.AllName; }, true);
121:             //llsAllNames.ItemsSource = DataSource;
122:
123:             var nameList2 = nameList.OrderBy(j => j.AllName).ToList() as
List<All>;
124:             //lstSearch.ItemsSource = nameList;
125:             for (int i = 0; i < nameList2.Count; i++)
126:             {
127:                 lstSearch.Items.Add(nameList2[i] as All);
128:             }
129:             //lstSearch.ItemTemplate.
130:             //lstSearch.DisplayMemberPath = "NoteName" + " (" + "CreationDate"
+ ")";
131:             lstSearch.DisplayMemberPath = "AllNameSource";
132:             MessageBox.Show(AppResources.SearchCompleted);
133:         }
134:     }
135: }
136:
137: private void lstSearch_SelectionChanged(object sender,
SelectionChangedEventArgs e)
138: {
139:     try
140:     {
141:         if (lstSearch.SelectedIndex != -1)
142:         {
143:             All selectedName = lstSearch.SelectedItem as All;
144:             NavigationService.Navigate(new Uri("/NameDetailPage.xaml#" +
selectedName.AllId, UriKind.Relative));
145:             lstSearch.SelectedIndex = -1;
146:         }
147:
148:     }
149:     catch (Exception)
150:     {
151:         MessageBox.Show(AppResources.SystemFault);

```

```

152:         }
153:     }
154:
155:     protected override void OnNavigatedTo(NavigationEventArgs e)
156:     {
157:         base.OnNavigatedTo(e);
158:     }
159:     protected override void OnNavigatedFrom(NavigationEventArgs e)
160:     {
161:         base.OnNavigatedFrom(e);
162:     }
163:
164:     private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
165:     {
166:         if (this.NavigationService.CanGoBack)
167:         {
168:             this.NavigationService.Navigate(new Uri("/MainPage.xaml",
UriKind.Relative));
169:         }
170:     }
171:
172:     private void txtSearch_KeyDown(object sender,
System.Windows.Input.KeyEventArgs e)
173:     {
174:         if (e.Key == Key.Enter)
175:         {
176:             lstSearch.Items.Clear();
177:             var nameList = new List<All>();
178:             if (txtSearch.Text.TrimStart().TrimEnd().Length < 1)
179:             {
180:                 MessageBox.Show(AppResources.SearchTrimFault);
181:             }
182:             else
183:             {
184:                 using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
185:                 {
186:                     if (cbSearchWithMeaning.IsChecked == true)
187:                     {
188:                         nameList = context.AllNames.Where(j =>
j.AllNameMeaning.ToLower().Contains(txtSearch.Text.TrimEnd().TrimStart().ToLower())).ToList()
189:                     }
190:                     else
191:                     {
192:                         nameList = context.AllNames.Where(j =>
j.AllName.ToLower().Contains(txtSearch.Text.TrimEnd().TrimStart().ToLower())).ToList() as
List<All>;
193:                     }
194:                     //var noteList = context.Notes.ToList() as List<Note>;
195:
196:                     if (nameList != null)
197:                     {
198:                         txtSearchResult.Text = AppResources.SearchResults + " (" +
nameList.Count() + ")";
199:                     }
200:
201:                     //List<All> allNames = nameList.OrderBy(j =>
j.AllName).ToList() as List<All>;
202:                     //List<AlphaKeyGroup<All>> DataSource =
AlphaKeyGroup<All>.CreateGroups(allNames,
203:                     // System.Threading.Thread.CurrentThread.CurrentUICulture,
204:                     // (All a) => { return a.AllName; }, true);
205:                     //IlsAllNames.ItemsSource = DataSource;
206:
207:                     var nameList2 = nameList.OrderBy(j => j.AllName).ToList() as
List<All>;

```

```
208:                                     //lstSearch.ItemsSource = nameList;
209:                                     for (int i = 0; i < nameList2.Count; i++)
210:                                     {
211:                                         lstSearch.Items.Add(nameList2[i] as All);
212:                                     }
213:                                     //lstSearch.ItemTemplate.
214:                                     //lstSearch.DisplayMemberPath = "NoteName" + " (" +
"CreationDate" + ")";
215:                                     lstSearch.DisplayMemberPath = "AllNameSource";
216:                                     MessageBox.Show(AppResources.SearchCompleted);
217:                                     }
218:                                 }
219:                             }
220:                         }
221:
222:         private void PhoneApplicationPage_Loaded(object sender, RoutedEventArgs e)
223:         {
224:             //txtSearch.Focus();
225:         }
226:     }
227: }
```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

1.2.29 StatisticsPage.xaml.cs

This is file StatisticsPage.xaml.cs.

Body Source

```
1: ?using System;
2: using System.Collections.Generic;
3: using System.IO;
4: using System.Linq;
5: using System.Net;
6: using System.Text;
7: using System.Windows;
8: using System.Windows.Controls;
9: using System.Windows.Controls.Primitives;
10: using System.Windows.Input;
11: using System.Windows.Media;
12: using System.Windows.Media.Imaging;
13: using System.Windows.Navigation;
14: using Microsoft.Phone.Tasks;
15: using AwesomeDictionary.Resources;
16: using Microsoft.Phone.Controls;
17: using Microsoft.Phone.Shell;
18:
19: namespace AwesomeDictionary
20: {
21:     public partial class StatisticsPage : PhoneApplicationPage
22:     {
23:         public StatisticsPage()
24:         {
25:             InitializeComponent();
26:             lblStatistics.Text = AppResources.Statistics;
27:             SetBackgroundColor();
28:         }
29:
30:         private void PhoneApplicationPage_BackKeyPress(object sender,
System.ComponentModel.CancelEventArgs e)
31:         {
```

```

32:         if (this.NavigationService.CanGoBack)
33:         {
34:             this.NavigationService.Navigate(new Uri("/MainPage.xaml",
UriKind.Relative));
35:         }
36:     }
37:
38:     private void SetBackgroundColor()
39:     {
40:         AppSettings appSettings = new AppSettings();
41:         using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
42:         {
43:             appSettings = context.AppSettings.First() as AppSettings;
44:         }
45:
46:         if (appSettings.AppBackgroundImage != null)
47:         {
48:             MemoryStream stream = new MemoryStream(appSettings.AppBackgroundImage);
49:             BitmapImage image = new BitmapImage();
50:             image.SetSource(stream);
51:             ImageBrush ib = new ImageBrush();
52:             ib.ImageSource = image;
53:             this.LayoutRoot.Background = ib;
54:         }
55:         else
56:         {
57:             switch (appSettings.AppBackgroundColor)
58:             {
59:                 case "BLA":
60:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
61:                     break;
62:                 case "BLU":
63:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Blue);
64:                     break;
65:                 case "BRO":
66:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Brown);
67:                     break;
68:                 case "RED":
69:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Red);
70:                     break;
71:                 case "GRE":
72:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Green);
73:                     break;
74:                 case "GRA":
75:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Gray);
76:                     break;
77:                 case "YEL":
78:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Yellow);
79:                     break;
80:                 case "ORA":
81:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Orange);
82:                     break;
83:                 case "PUR":
84:                     this.LayoutRoot.Background = new
SolidColorBrush(Colors.Purple);
85:                     break;
86:                 default:
87:                     this.LayoutRoot.Background = new SolidColorBrush(Colors.Black);
88:                     break;
89:             }
90:         }
91:     }
92:
93:     protected override void OnNavigatedTo(NavigationEventArgs e)
94:     {
95:         base.OnNavigatedTo(e);

```

```

96:         SetStatistic();
97:     }
98:
99:     protected override void OnNavigatedFrom(NavigationEventArgs e)
100:    {
101:        base.OnNavigatedFrom(e);
102:        //while (NavigationService.CanGoBack)
103:        //NavigationService.RemoveBackEntry();
104:    }
105:
106:
107:     private void SetStatistic()
108:    {
109:        StringBuilder sb = new StringBuilder();
110:        int allCount, favouriteCount;
111:
112:
113:        using (var context = new
AwesomeDictionaryDataContext(AwesomeDictionaryDataContext.ConnectionString))
114:        {
115:            favouriteCount = context.Favourites.ToList().Count;
116:            allCount = context.AllNames.ToList().Count;
117:        }
118:
119:        sb.AppendLine(AppResources.TotalNameCount + ": " + allCount);
120:        sb.AppendLine(AppResources.FavouriteNameCount + ": " + favouriteCount);
121:        //sb.AppendLine(AppResources.MostListenArtist + ": " + artistName);
122:        //sb.AppendLine(AppResources.MostListenLabel + ": " + labelName);
123:        //sb.AppendLine(AppResources.BestAlbum + ": " + bestAlbum);
124:        //sb.AppendLine(AppResources.WorstAlbum + ": " + worstAlbum);
125:
126:        txtStatistics.Text = sb.ToString();
127:        txtStatistics.IsReadOnly = true;
128:    }
129: }
130: }

```

Namespaces

Name	Description
AwesomeDictionary (see page 1)	This is namespace AwesomeDictionary.

Index

A

AboutPage class 17

about AboutPage class 17

AboutPage 17

AboutPage.xaml.cs 60

All class 18

about All class 18

AllId 18

AllMeaning 18

AllName 19

AllNameMeaning 19

AllNameSource 19

AllSource 19

All.cs 63

AlmancaTurkce class 19

about AlmancaTurkce class 19

AlmancaTurkceld 20

AlmancaTurkceMeaning 20

AlmancaTurkceName 20

AlmancaTurkceNameMeaning 20

AlmancaTurkce.cs 64

AlphaKeyGroup class 21

about AlphaKeyGroup class 21

AlphaKeyGroup 21

CreateGroups 21

GetKeyDelegate 23

Key 22

AlphaKeyGroup.cs 64

App class 23

about App class 23

App 23

ReadFile 24

RootFrame 25

App.xaml.cs 66

AppResources class 2

About 6

about AppResources class 2

AboutAwesomeDictionary 6

AboutTheApp 6

AboutTheAppText 6

AddToFavourites 6

AppResources 5

AtLeastOneDictionary 6

Background 6

BackgroundColor 6

BackgroundColorChangedSuccessfully 6

BackgroundImage 7

BackgroundImageChangedSuccessfully 7

BackgroundImageRemovedSuccessfully 7

BackgroundSettingsResetSuccessfully 7

Black 7

Blue 7

Brown 7

BuyukLugat 7

Cancel 7

ComputerDictionary 8

ContactWithUs 8

Culture 8

DictionariesInstalledSuccessfully 8

DictionariesUninstalledSuccessfully 8

DictionaryInstall 8

English 8

EnglishTurkishVol1 8

EnglishTurkishVol2 9

ExitApp 9

ExitAppQuestion 9

FavouriteNameCount 9

Font 9

FontFamily 9

FontFamilyChangedSuccessfully 9

FontSize 9

FontSizeChangedSuccessfully 9

GeneralSettings 10

German 10

GermanTurkish 10

Gray 10

Green 10

Install 10

Installing 10

Language 10	Turkish 15
LanguageWarning 10	Uninstall 15
Meaning 11	Word 15
MyFavourites 11	WordAddedFavouriteSuccessfully 15
Ok 11	WordAlreadyFavourite 15
Orange 11	WordAndMeaning 15
OxfordDictionary 11	WordMeaning 16
Purple 11	WordRemovedFavouriteSuccessfully 16
RandomWords 11	Yellow 16
Rate 11	AppResources.Designer.cs 73
Red 11	AppSettings class 25
RemoveBackgroundImage 12	about AppSettings class 25
RemoveFromFavourite 12	AppBackgroundColor 25
RemoveFromFavouriteQuestion 12	AppBackgroundImage 26
RemoveFromFavourites 12	AppLangName 26
ResetSettings 12	AppSettingsId 26
ResourceFlowDirection 12	FontFamily 26
ResourceLanguage 12	FontSize 26
ResourceManager 12	AppSettings.cs 87
RisaleNur 12	AssemblyInfo.cs 87
Search 13	AwesomeDictionary 1
SearchCompleted 13	AwesomeDictionary namespace 1
SearchInMeanings 13	Classes 16
SearchResults 13	AwesomeDictionary.AboutPage 17
SearchTrimFault 13	AwesomeDictionary.AboutPage.AboutPage 17
Select 13	AwesomeDictionary.All 18
SelectBackgroundColor 13	AwesomeDictionary.All.AllId 18
Selected 13	AwesomeDictionary.All.AllMeaning 18
SelectFontFamily 13	AwesomeDictionary.All.AllName 19
SelectFontSize 14	AwesomeDictionary.All.AllNameMeaning 19
SelectLanguage 14	AwesomeDictionary.All.AllNameSource 19
SendWithAwesomeDictionaryApp 14	AwesomeDictionary.All.AllSource 19
SendWithEmail 14	AwesomeDictionary.AlmancaTurkce 19
SendWithSMS 14	AwesomeDictionary.AlmancaTurkce.AlmancaTurkceId 20
Settings 14	AwesomeDictionary.AlmancaTurkce.AlmancaTurkceMeaning 20
Share 14	AwesomeDictionary.AlmancaTurkce.AlmancaTurkceName 20
Source 14	AwesomeDictionary.AlmancaTurkce.AlmancaTurkceNameMeaning 20
Statistics 14	
Synchronizing 15	AwesomeDictionary.AlphaKeyGroup 21
SystemFault 15	AwesomeDictionary.AlphaKeyGroup.AlphaKeyGroup 21
TotalNameCount 15	

AwesomeDictionary.AlphaKeyGroup.CreateGroups 21	AwesomeDictionary.AwesomeDictionaryDataContext.RisaleNurs 29
AwesomeDictionary.AlphaKeyGroup.GetKeyDelegate 23	
AwesomeDictionary.AlphaKeyGroup.Key 22	AwesomeDictionary.BackgroundColorSettingsPage 29
AwesomeDictionary.App 23	AwesomeDictionary.BackgroundColorSettingsPage.BackgroundColorSettingsPage 30
AwesomeDictionary.App.App 23	AwesomeDictionary.BackgroundColorSettingsPage.OnFragmentNavigation 30
AwesomeDictionary.App.ReadFile 24	
AwesomeDictionary.App.RootFrame 25	AwesomeDictionary.BackgroundColorSettingsPage.OnNavigatedFrom 31
AwesomeDictionary.AppSettings 25	AwesomeDictionary.BackgroundColorSettingsPage.OnNavigatedTo 31
AwesomeDictionary.AppSettings.AppBackgroundColor 25	
AwesomeDictionary.AppSettings.AppBackgroundImage 26	AwesomeDictionary.BilisimSozlugu 31
AwesomeDictionary.AppSettings.AppLangName 26	AwesomeDictionary.BilisimSozlugu.BilisimSozluguld 32
AwesomeDictionary.AppSettings.AppSettingsId 26	AwesomeDictionary.BilisimSozlugu.BilisimSozluguMeaning 32
AwesomeDictionary.AppSettings.FontFamily 26	AwesomeDictionary.BilisimSozlugu.BilisimSozluguName 32
AwesomeDictionary.AppSettings.FontSize 26	AwesomeDictionary.BilisimSozlugu.BilisimSozluguNameMeaning 32
AwesomeDictionary.AwesomeDictionaryDataContext 26	
AwesomeDictionary.AwesomeDictionaryDataContext.AllNames 27	AwesomeDictionary.BuyukLugat 33
AwesomeDictionary.AwesomeDictionaryDataContext.AlmancaTurkces 28	AwesomeDictionary.BuyukLugat.BuyukLugatlId 33
AwesomeDictionary.AwesomeDictionaryDataContext.AppSettings 28	AwesomeDictionary.BuyukLugat.BuyukLugatMeaning 33
AwesomeDictionary.AwesomeDictionaryDataContext.AwesomeDictionaryDataContext 27	AwesomeDictionary.BuyukLugat.BuyukLugatName 33
AwesomeDictionary.AwesomeDictionaryDataContext.Bilisims 28	AwesomeDictionary.BuyukLugat.BuyukLugatNameMeaning 34
AwesomeDictionary.AwesomeDictionaryDataContext.BuyukLugats 28	AwesomeDictionary.csproj 88
AwesomeDictionary.AwesomeDictionaryDataContext.ConnectionString 28	AwesomeDictionary.EnglishTurkishVol1 34
AwesomeDictionary.AwesomeDictionaryDataContext.EnglishTurkishVol1s 28	AwesomeDictionary.EnglishTurkishVol1.EnglishVol1Id 34
AwesomeDictionary.AwesomeDictionaryDataContext.EnglishTurkishVol2s 28	AwesomeDictionary.EnglishTurkishVol1.EnglishVol1Meaning 34
AwesomeDictionary.AwesomeDictionaryDataContext.Favourites 29	AwesomeDictionary.EnglishTurkishVol1.EnglishVol1Name 35
AwesomeDictionary.AwesomeDictionaryDataContext.Kelimes 29	AwesomeDictionary.EnglishTurkishVol1.EnglishVol1NameMeaning 35
AwesomeDictionary.AwesomeDictionaryDataContext.Oxfords 29	AwesomeDictionary.EnglishTurkishVol2 35
	AwesomeDictionary.EnglishTurkishVol2.EnglishVol2Id 35
	AwesomeDictionary.EnglishTurkishVol2.EnglishVol2Meaning 36
	AwesomeDictionary.EnglishTurkishVol2.EnglishVol2Name 36
	AwesomeDictionary.EnglishTurkishVol2.EnglishVol2NameMeaning 36
	AwesomeDictionary.Favourite 36

AwesomeDictionary.Favourite.FavouriteAllId 37	m
AwesomeDictionary.Favourite.FavouriteId 37	48
AwesomeDictionary.Favourite.FavouriteName 37	AwesomeDictionary.LanguageSettingsPage.OnNavigatedTo 48
AwesomeDictionary.FontFamilySettingsPage 37	AwesomeDictionary.LocalizedStrings 49
AwesomeDictionary.FontFamilySettingsPage.FontFamilySettingsPage 38	AwesomeDictionary.LocalizedStrings.LocalizedResources 49
AwesomeDictionary.FontFamilySettingsPage.OnFragmentNavigation 38	AwesomeDictionary.MainPage 49
AwesomeDictionary.FontFamilySettingsPage.OnNavigatedFrom 39	AwesomeDictionary.MainPage.MainPage 49
AwesomeDictionary.FontFamilySettingsPage.OnNavigatedTo 39	AwesomeDictionary.NameDetailPage 50
AwesomeDictionary.FontSizeSettingsPage 39	AwesomeDictionary.NameDetailPage.flag 52
AwesomeDictionary.FontSizeSettingsPage.FontSizeSettingsPage 40	AwesomeDictionary.NameDetailPage.isFilled 52
AwesomeDictionary.FontSizeSettingsPage.OnFragmentNavigation 40	AwesomeDictionary.NameDetailPage.NameDetailPage 51
AwesomeDictionary.FontSizeSettingsPage.OnNavigatedFrom 41	AwesomeDictionary.NameDetailPage.OnFragmentNavigation 53
AwesomeDictionary.FontSizeSettingsPage.OnNavigatedTo 41	AwesomeDictionary.NameDetailPage.OnNavigatedFrom 54
AwesomeDictionary.GeneralSettingsPage 41	AwesomeDictionary.NameDetailPage.OnNavigatedTo 54
AwesomeDictionary.GeneralSettingsPage.GeneralSettingsPage 42	AwesomeDictionary.NameDetailPage.pageName 52
AwesomeDictionary.GeneralSettingsPage.OnNavigatedFrom 45	AwesomeDictionary.NameDetailPage.ratingValue 52
AwesomeDictionary.GeneralSettingsPage.OnNavigatedTo 45	AwesomeDictionary.NameDetailPage.wordId 52
AwesomeDictionary.GeneralSettingsPage.ReadFile 45	AwesomeDictionary.OxfordEnglishEnglish 54
AwesomeDictionary.KelimeAnlamlari 46	AwesomeDictionary.OxfordEnglishEnglish.OxfordId 55
AwesomeDictionary.KelimeAnlamlari.KelimeAnlamlariId 46	AwesomeDictionary.OxfordEnglishEnglish.OxfordMeaning 55
AwesomeDictionary.KelimeAnlamlari.KelimeAnlamlariMeaning 46	AwesomeDictionary.OxfordEnglishEnglish.OxfordName 55
AwesomeDictionary.KelimeAnlamlari.KelimeAnlamlariName 47	AwesomeDictionary.OxfordEnglishEnglish.OxfordNameMeaning 55
AwesomeDictionary.KelimeAnlamlari.KelimeAnlamlariNameMeaning 47	AwesomeDictionary.Resources 2
AwesomeDictionary.LanguageSettingsPage 47	AwesomeDictionary.Resources namespace 2
AwesomeDictionary.LanguageSettingsPage.LanguageSettingsPage 47	Classes 2
AwesomeDictionary.LanguageSettingsPage.OnNavigatedFrom 47	AwesomeDictionary.Resources.AppResources 2
	AwesomeDictionary.Resources.AppResources.About 6
	AwesomeDictionary.Resources.AppResources.AboutAwesomeDictionary 6
	AwesomeDictionary.Resources.AppResources.AboutTheApp 6
	AwesomeDictionary.Resources.AppResources.AboutTheAppText 6
	AwesomeDictionary.Resources.AppResources.AddToFavourites 6
	AwesomeDictionary.Resources.AppResources.AppResources 5
	AwesomeDictionary.Resources.AppResources.AtLeastOneDictionary 5

ctionary 6	AwesomeDictionary.Resources.AppResources.ExitAppQuesti on 9
AwesomeDictionary.Resources.AppResources.Background 6	
AwesomeDictionary.Resources.AppResources.BackgroundCo lor 6	AwesomeDictionary.Resources.AppResources.FavouriteNam eCount 9
AwesomeDictionary.Resources.AppResources.BackgroundCo lorChangedSuccessfully 6	AwesomeDictionary.Resources.AppResources.Font 9
AwesomeDictionary.Resources.AppResources.BackgroundIm age 7	AwesomeDictionary.Resources.AppResources.FontFamily 9
AwesomeDictionary.Resources.AppResources.BackgroundIm ageChangedSuccessfully 7	AwesomeDictionary.Resources.AppResources.FontFamilyCh angedSuccessfully 9
AwesomeDictionary.Resources.AppResources.BackgroundIm ageRemovedSuccessfully 7	AwesomeDictionary.Resources.AppResources.FontSize 9
AwesomeDictionary.Resources.AppResources.BackgroundIm ageResetSuccessfully 7	AwesomeDictionary.Resources.AppResources.FontSizeChan gedSuccessfully 9
AwesomeDictionary.Resources.AppResources.Black 7	AwesomeDictionary.Resources.AppResources.GeneralSetting s 10
AwesomeDictionary.Resources.AppResources.Blue 7	AwesomeDictionary.Resources.AppResources.German 10
AwesomeDictionary.Resources.AppResources.Brown 7	AwesomeDictionary.Resources.AppResources.GermanTurkis h 10
AwesomeDictionary.Resources.AppResources.BuyukLugat 7	AwesomeDictionary.Resources.AppResources.Gray 10
AwesomeDictionary.Resources.AppResources.Cancel 7	AwesomeDictionary.Resources.AppResources.Green 10
AwesomeDictionary.Resources.AppResources.ComputerDicti onary 8	AwesomeDictionary.Resources.AppResources.Install 10
AwesomeDictionary.Resources.AppResources.ContactWithUs 8	AwesomeDictionary.Resources.AppResources.Installing 10
AwesomeDictionary.Resources.AppResources.Culture 8	AwesomeDictionary.Resources.AppResources.Language 10
AwesomeDictionary.Resources.AppResources.DictionariesIns talledSuccessfully 8	AwesomeDictionary.Resources.AppResources.LanguageWar ning 10
AwesomeDictionary.Resources.AppResources.DictionariesUn installedSuccessfully 8	AwesomeDictionary.Resources.AppResources.Meaning 11
AwesomeDictionary.Resources.AppResources.DictionaryInsta ll 8	AwesomeDictionary.Resources.AppResources.MyFavourites 11
AwesomeDictionary.Resources.AppResources.English 8	AwesomeDictionary.Resources.AppResources.Ok 11
AwesomeDictionary.Resources.AppResources.EnglishTurkish Vol1 8	AwesomeDictionary.Resources.AppResources.Orange 11
AwesomeDictionary.Resources.AppResources.EnglishTurkish Vol2 9	AwesomeDictionary.Resources.AppResources.OxfordDictiona ry 11
AwesomeDictionary.Resources.AppResources.ExitApp 9	AwesomeDictionary.Resources.AppResources.Purple 11
	AwesomeDictionary.Resources.AppResources.RandomWords 11
	AwesomeDictionary.Resources.AppResources.Rate 11
	AwesomeDictionary.Resources.AppResources.Red 11
	AwesomeDictionary.Resources.AppResources.RemoveBackg roundImage 12
	AwesomeDictionary.Resources.AppResources.RemoveFromF

avourite 12	14
AwesomeDictionary.Resources.AppResources.RemoveFromFavouriteQuestion 12	AwesomeDictionary.Resources.AppResources.SendWithSMS 14
AwesomeDictionary.Resources.AppResources.RemoveFromFavourites 12	AwesomeDictionary.Resources.AppResources.Settings 14
AwesomeDictionary.Resources.AppResources.ResetSettings 12	AwesomeDictionary.Resources.AppResources.Share 14
AwesomeDictionary.Resources.AppResources.ResourceFlowDirection 12	AwesomeDictionary.Resources.AppResources.Source 14
AwesomeDictionary.Resources.AppResources.ResourceLanguage 12	AwesomeDictionary.Resources.AppResources.Statistics 14
AwesomeDictionary.Resources.AppResources.ResourceManager 12	AwesomeDictionary.Resources.AppResources.Synchronizing 15
AwesomeDictionary.Resources.AppResources.RisaleNur 12	AwesomeDictionary.Resources.AppResources.SystemFault 15
AwesomeDictionary.Resources.AppResources.Search 13	AwesomeDictionary.Resources.AppResources.TotalNameCount 15
AwesomeDictionary.Resources.AppResources.SearchCompleted 13	AwesomeDictionary.Resources.AppResources.Turkish 15
AwesomeDictionary.Resources.AppResources.SearchInMeanings 13	AwesomeDictionary.Resources.AppResources.Uninstall 15
AwesomeDictionary.Resources.AppResources.SearchResults 13	AwesomeDictionary.Resources.AppResources.Word 15
AwesomeDictionary.Resources.AppResources.SearchTrimFault 13	AwesomeDictionary.Resources.AppResources.WordAddedFavouriteSuccessfully 15
AwesomeDictionary.Resources.AppResources.Select 13	AwesomeDictionary.Resources.AppResources.WordAlreadyFavourite 15
AwesomeDictionary.Resources.AppResources.SelectBackgroundAndColor 13	AwesomeDictionary.Resources.AppResources.WordAndMeaning 15
AwesomeDictionary.Resources.AppResources.Selected 13	AwesomeDictionary.Resources.AppResources.WordMeaning 16
AwesomeDictionary.Resources.AppResources.SelectFontFamily 13	AwesomeDictionary.Resources.AppResources.WordRemovedFavouriteSuccessfully 16
AwesomeDictionary.Resources.AppResources.SelectFontSize 14	AwesomeDictionary.Resources.AppResources.Yellow 16
AwesomeDictionary.Resources.AppResources.SelectLanguage 14	AwesomeDictionary.RisaleNur 56
AwesomeDictionary.Resources.AppResources.SendWithAwesomeDictionaryApp 14	AwesomeDictionary.RisaleNur.RisaleNurId 56
AwesomeDictionary.Resources.AppResources.SendWithEmail 1	AwesomeDictionary.RisaleNur.RisaleNurMeaning 56
	AwesomeDictionary.RisaleNur.RisaleNurName 56
	AwesomeDictionary.RisaleNur.RisaleNurNameMeaning 57
	AwesomeDictionary.SearchPage 57
	AwesomeDictionary.SearchPage.OnNavigatedFrom 58
	AwesomeDictionary.SearchPage.OnNavigatedTo 58
	AwesomeDictionary.SearchPage.SearchPage 57
	AwesomeDictionary.sln 88
	AwesomeDictionary.StatisticsPage 58
	AwesomeDictionary.StatisticsPage.OnNavigatedFrom 59
	AwesomeDictionary.StatisticsPage.OnNavigatedTo 59

AwesomeDictionary.StatisticsPage.StatisticsPage 59

AwesomeDictionaryDataContext class 26

about AwesomeDictionaryDataContext class 26

AllNames 27

AlmancaTurkces 28

AppSettings 28

AwesomeDictionaryDataContext 27

Bilisims 28

BuyukLugats 28

ConnectionString 28

EnglishTurkishVol1s 28

EnglishTurkishVol2s 28

Favourites 29

Kelimes 29

Oxfords 29

RisaleNurs 29

AwesomeDictionaryDataContext.cs 88

B

BackgroundColorSettingsPage class 29

about BackgroundColorSettingsPage class 29

BackgroundColorSettingsPage 30

OnFragmentNavigation 30

OnNavigatedFrom 31

OnNavigatedTo 31

BackgroundColorSettingsPage.xaml.cs 89

BilirimSozlugu class 31

about BilirimSozlugu class 31

BilirimSozluguld 32

BilirimSozluguMeaning 32

BilirimSozluguName 32

BilirimSozluguNameMeaning 32

BilirimSozlugu.cs 92

BuyukLugat class 33

about BuyukLugat class 33

BuyukLugatlId 33

BuyukLugatMeaning 33

BuyukLugatName 33

BuyukLugatNameMeaning 34

BuyukLugat.cs 93

E

EnglishTurkishVol1 class 34

about EnglishTurkishVol1 class 34

EnglishVol1Id 34

EnglishVol1Meaning 34

EnglishVol1Name 35

EnglishVol1NameMeaning 35

EnglishTurkishVol1.cs 94

EnglishTurkishVol2 class 35

about EnglishTurkishVol2 class 35

EnglishVol2Id 35

EnglishVol2Meaning 36

EnglishVol2Name 36

EnglishVol2NameMeaning 36

EnglishTurkishVol2.cs 94

F

Favourite class 36

about Favourite class 36

FavouriteAllId 37

FavouriteId 37

FavouriteName 37

Favourite.cs 95

Files 59

FontFamilySettingsPage class 37

about FontFamilySettingsPage class 37

FontFamilySettingsPage 38

OnFragmentNavigation 38

OnNavigatedFrom 39

OnNavigatedTo 39

FontFamilySettingsPage.xaml.cs 96

FontSizeSettingsPage class 39

about FontSizeSettingsPage class 39

FontSizeSettingsPage 40

OnFragmentNavigation 40

OnNavigatedFrom 41

OnNavigatedTo 41

FontSizeSettingsPage.xaml.cs 98

G

GeneralSettingsPage class 41
 about GeneralSettingsPage class 41
 GeneralSettingsPage 42
 OnNavigatedFrom 45
 OnNavigatedTo 45
 ReadFile 45
GeneralSettingsPage.xaml.cs 101

K

KelimeAnlamlari class 46
 about KelimeAnlamlari class 46
 KelimeAnlamlarId 46
 KelimeAnlamlariMeaning 46
 KelimeAnlamlariName 47
 KelimeAnlamlariNameMeaning 47
KelimeAnlamlari.cs 120

L

LanguageSettingsPage class 47
 about LanguageSettingsPage class 47
 LanguageSettingsPage 47
 OnNavigatedFrom 48
 OnNavigatedTo 48
LanguageSettingsPage.xaml.cs 121
LocalizedStrings class 49
 about LocalizedStrings class 49
 LocalizedResources 49
LocalizedStrings.cs 125

M

MainPage class 49
 about MainPage class 49
 MainPage 49
MainPage.xaml.cs 125

N

NameDetailPage class 50

 about NameDetailPage class 50
 flag 52
 isFilled 52
 NameDetailPage 51
 OnFragmentNavigation 53
 OnNavigatedFrom 54
 OnNavigatedTo 54
 pageName 52
 ratingValue 52
 wordId 52

NameDetailPage.xaml.cs 129

O

OxfordEnglishEnglish class 54
 about OxfordEnglishEnglish class 54
 OxfordId 55
 OxfordMeaning 55
 OxfordName 55
 OxfordNameMeaning 55
OxfordEnglishEnglish.cs 139

R

RisaleNur class 56
 about RisaleNur class 56
 RisaleNurId 56
 RisaleNurMeaning 56
 RisaleNurName 56
 RisaleNurNameMeaning 57
RisaleNur.cs 139

S

SearchPage class 57
 about SearchPage class 57
 OnNavigatedFrom 58
 OnNavigatedTo 58
 SearchPage 57
SearchPage.xaml.cs 140
StatisticsPage class 58
 about StatisticsPage class 58
 OnNavigatedFrom 59

OnNavigatedTo 59

StatisticsPage 59

StatisticsPage.xaml.cs 144