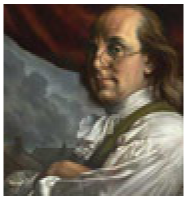
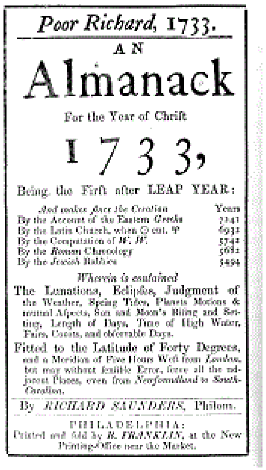
Tip of the Day or Phrases from Poor Richard’s Almanac…an AI2 Tutorial demonstrating a way to use TTS and Yandex Translate in an app

This tutorial about the framework for a multilingual ‘Tip of the Day’ app that started out simply. An AI2 novice wanted to develop an app that would present random phrases from one of his favorite writers. I started playing with some blocks. The playing sort of got out of hand.



Poor Richard’s Almanac was a published, written and printed by Benjamin Franklin in the late 1700’s in Colonial America. The almanac contained lots of “words of wisdom” for its readers. Mr. Franklin was a printer, a philosopher and one of America’s foremost statesman. His almanac contains numerous tidbits of Colonial philosophy.

Poor Richard is the name of the Tip of the Day example app you can create in this tutorial. The app takes a few old English sayings from Mr. Franklin’s almanac and presents the sayings in various languages (or in the original). The translations may be amusing to non-English speakers. Certainly some of the old English sentence structure is also amusing to modern English speakers. Sometimes, the translation service gets the intent of the original English correct.



Poor Richard, 1733.

AN

Almanack

For the Year of Chriſt

1733,

Being the Firſt after LEAP YEAR:

And makes ſince the Creation	Years
By the Account of the Eastern *Greeks*	7241
By the Latin Church, when ☉ ent. ♈	6932
By the Computation of *W. W.*	5742
By the *Roman* Chronology	5682
By the *Jewiſh* Rabbies	5494

Wherein is contained

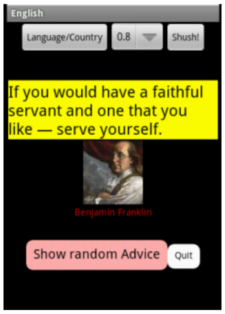
The Lunations, Eclipſes, Judgment of the Weather, Spring Tides, Planets Motions & mutual Aſpects, Sun and Moon’s Riſing and Setting, Length of Days, Time of High Water, Fairs, Courts, and obſervable Days.

Fitted to the Latitude of Forty Degrees, and a Meridian of Five Hours Weſt from *London*, but may without ſenſible Error, ſerve all the adjacent Places, even from *Newfoundland* to *South-Carolina*.

By *RICHARD SAUNDERS*, Philom.

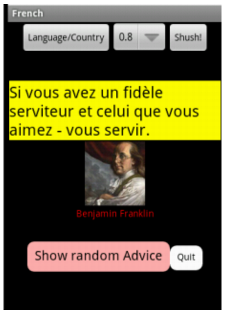
PHILADELPHIA:
Printed and ſold by *B. FRANKLIN*, at the New Printing-Office near the Market.

Poor Richard displays Franklin’s wisdom in English, or the selected language and “reads” the advice out loud.



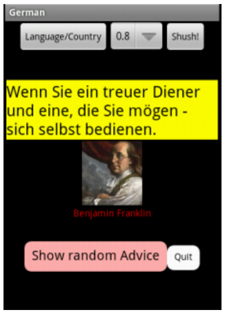
If you would have a faithful servant and one that you like — serve yourself.

English



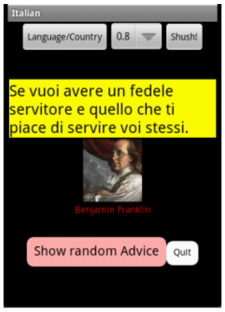
Si vous avez un fidèle serviteur et celui que vous aimez - vous servir.

French



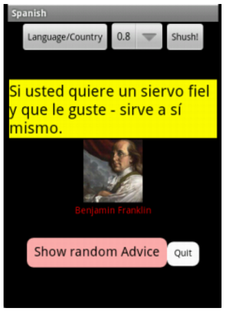
Wenn Sie ein treuer Diener und eine, die Sie mögen - sich selbst bedienen.

German



Se vuoi avere un fedele servitore e quello che ti piace di servire voi stessi.

Italian

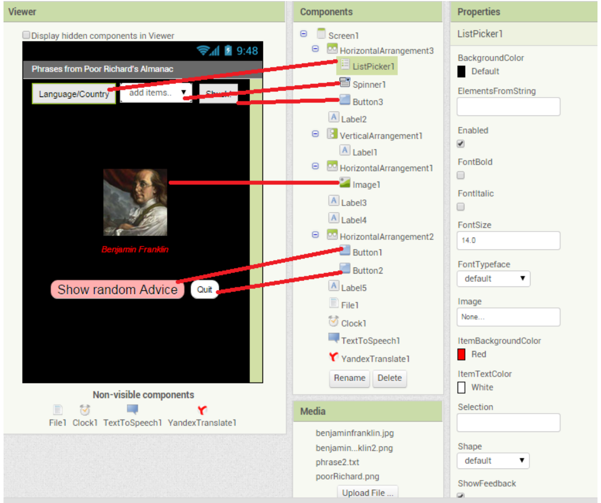


Si usted quiere un siervo fiel y que le guste - sirve a sí mismo.

Spanish

Well, ok, the translation is not perfect.

The Designer

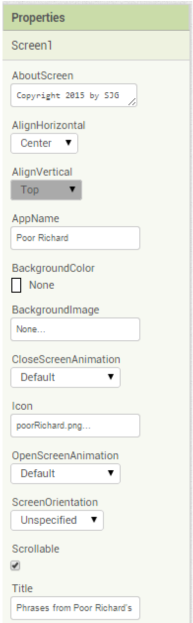


- Screen1
  - HorizontalArrangement3
    - ListPicker1
    - Spinner1
    - Button3
  - Label2
  - VerticalArrangement1
    - Label1
  - HorizontalArrangement1
    - Image1
  - Label3
  - Label4
  - HorizontalArrangement2
    - Button1
    - Button2
  - Label5
  - File1
  - Clock1
  - TextToSpeech1
  - YandexTranslate1

Media:
benjaminfranklin.jpg
benjamin...klin2.png
phrase2.txt
poorRichard.png

ListPicker1
BackgroundColor: ■ Default
ElementsFromString:
Enabled: ☑
FontBold: □
FontItalic: □
FontSize: 14.0
FontTypeface: default
Image: None...
ItemBackgroundColor: 🟥 Red
ItemTextColor: □ White
Selection:
Shape: default
ShowFeedback: ☑

The Designer Screen



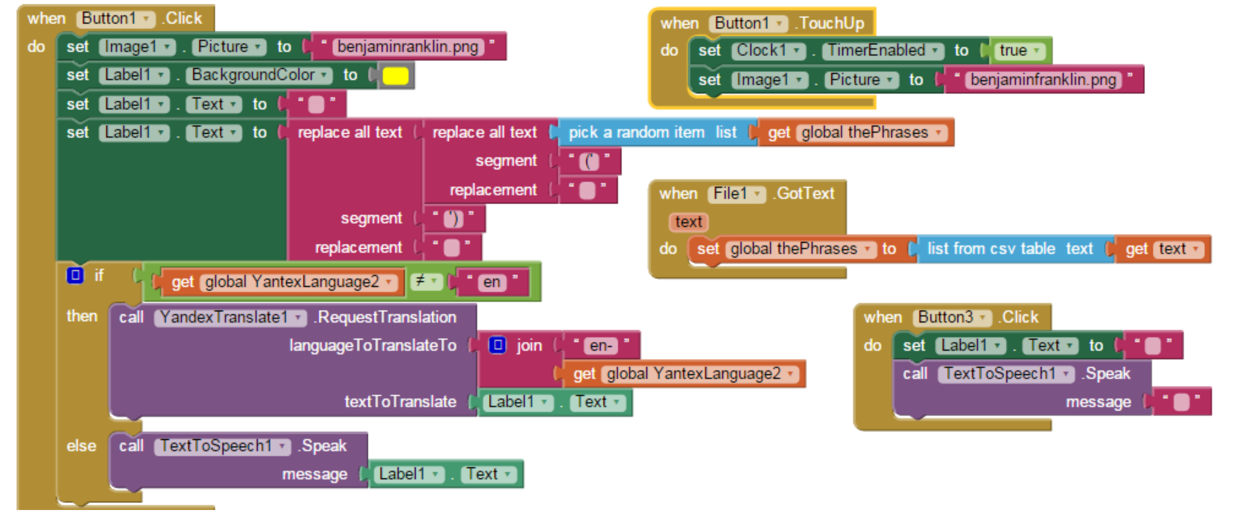
Screen1
AboutScreen: Copyright 2015 by SJG
AlignHorizontal: Center
AlignVertical: Top
AppName: Poor Richard
BackgroundColor: □ None
BackgroundImage: None...
CloseScreenAnimation: Default
Icon: poorRichard.png...
OpenScreenAnimation: Default
ScreenOrientation: Unspecified
Scrollable: ☑
Title: Phrases from Poor Richard's

Screen1 Properties Settings

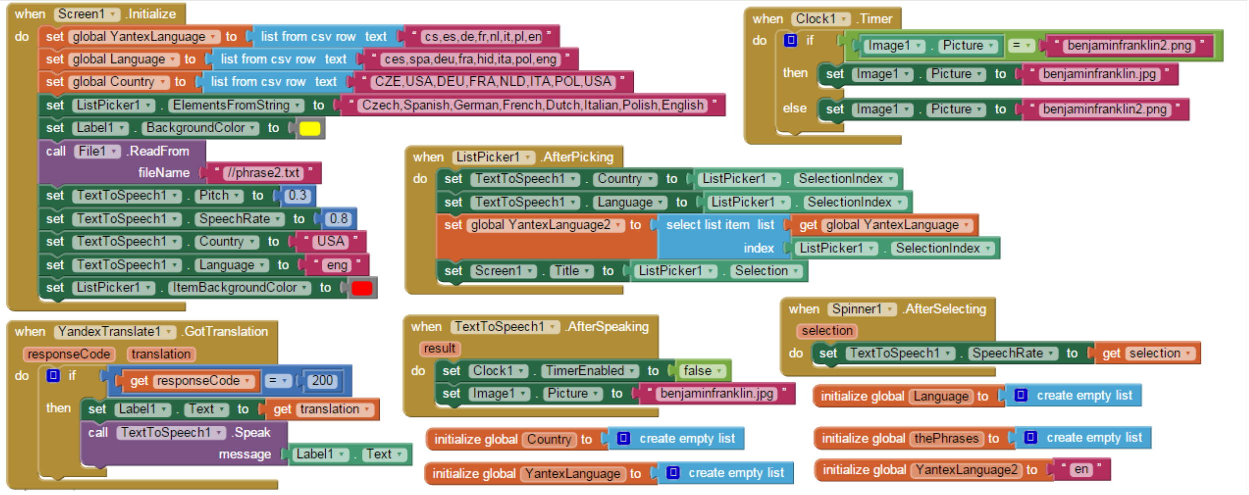
Just put the controls where you want them. The image above might help you decide where to place some of the controls.

The Blocks

All the blocks are here needed to create the app are here:



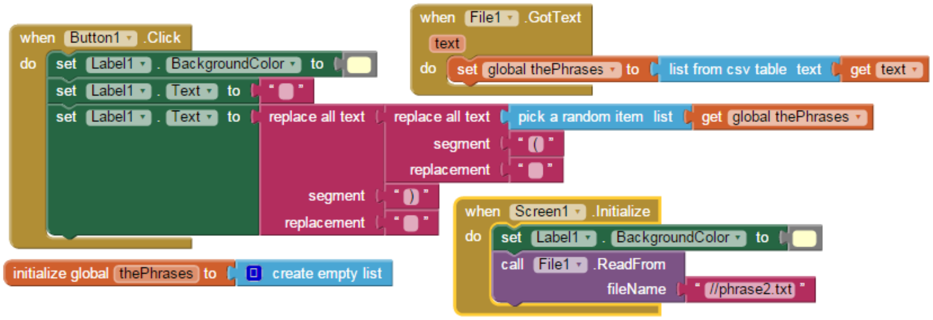
```
when Button1 .Click
do  set Image1 . Picture to " benjaminranklin.png "
    set Label1 . BackgroundColor to 🟨
    set Label1 . Text to " "
    set Label1 . Text to replace all text  replace all text  pick a random item list  get global thePhrases
                                                             segment  " (' "
                                                             replacement  " "
                         segment  " ') "
                         replacement  " "
    if  get global YantexLanguage2 ≠ " en "
    then  call YandexTranslate1 .RequestTranslation
                  languageToTranslateTo  join  " en- "
                                               get global YantexLanguage2
                  textToTranslate  Label1 . Text
    else  call TextToSpeech1 .Speak
                  message  Label1 . Text

when Button1 .TouchUp
do  set Clock1 . TimerEnabled to true
    set Image1 . Picture to " benjaminfranklin.png "

when File1 .GotText
  text
do  set global thePhrases to list from csv table text  get text

when Button3 .Click
do  set Label1 . Text to " "
    call TextToSpeech1 .Speak
            message  " "
```



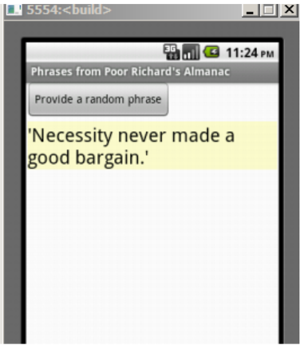
```
when Screen1 .Initialize
do  set global YantexLanguage to list from csv row text " cs,es,de,fr,nl,it,pl,en "
    set global Language to list from csv row text " ces,spa,deu,fra,hid,ita,pol,eng "
    set global Country to list from csv row text " CZE,USA,DEU,FRA,NLD,ITA,POL,USA "
    set ListPicker1 . ElementsFromString to " Czech,Spanish,German,French,Dutch,Italian,Polish,English "
    set Label1 . BackgroundColor to 🟨
    call File1 .ReadFrom
            fileName " //phrase2.txt "
    set TextToSpeech1 . Pitch to 0.3
    set TextToSpeech1 . SpeechRate to 0.8
    set TextToSpeech1 . Country to " USA "
    set TextToSpeech1 . Language to " eng "
    set ListPicker1 . ItemBackgroundColor to 🟥

when Clock1 .Timer
do  if  Image1 . Picture = " benjaminfranklin2.png "
    then  set Image1 . Picture to " benjaminfranklin.jpg "
    else  set Image1 . Picture to " benjaminfranklin2.png "

when ListPicker1 .AfterPicking
do  set TextToSpeech1 . Country to ListPicker1 . SelectionIndex
    set TextToSpeech1 . Language to ListPicker1 . SelectionIndex
    set global YantexLanguage2 to select list item list  get global YantexLanguage
                                                   index ListPicker1 . SelectionIndex
    set Screen1 . Title to ListPicker1 . Selection

when YandexTranslate1 .GotTranslation
  responseCode  translation
do  if  get responseCode = 200
    then  set Label1 . Text to get translation
          call TextToSpeech1 .Speak
                  message  Label1 . Text

when TextToSpeech1 .AfterSpeaking
  result
do  set Clock1 . TimerEnabled to false
    set Image1 . Picture to " benjaminfranklin.jpg "

when Spinner1 .AfterSelecting
  selection
do  set TextToSpeech1 . SpeechRate to get selection

initialize global Language to create empty list
initialize global Country to create empty list
initialize global thePhrases to create empty list
initialize global YantexLanguage to create empty list
initialize global YantexLanguage2 to " en "
```

The text file, containing the tips of the day, is part of the included aia file.   If you want to code the entire app, you will need to copy the text file and the images from the aia Media.

The image of the original app I coded shows how the phrase2.txt text file should be structured (note the misspelling (pharse2) on the example image ...sorry). The text file contains the “tips of the day” or advice material and is stored in Media. The text file is imported into the *thePhrases* List where a selection of Mr. Franklin’s witticisms become List items. So, if you do not want Benjamin talking, include your own text and images.



```
when Button1 .Click
do  set Label1 . BackgroundColor to
    set Label1 . Text to " "
    set Label1 . Text to  replace all text  replace all text  pick a random item  list  get global thePhrases
                                            segment      " ( "
                                            replacement  " "
                          segment      " ) "
                          replacement  " "

initialize global thePhrases to  create empty list

when File1 .GotText
 text
do  set global thePhrases to  list from csv table  text  get text

when Screen1 .Initialize
do  set Label1 . BackgroundColor to
    call File1 .ReadFrom
              fileName  " //phrase2.txt "
```

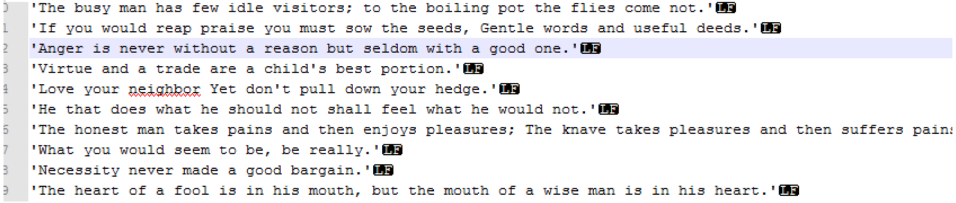


Phrases from Poor Richard's Almanac

Provide a random phrase

'Necessity never made a good bargain.'

The file below is called pharse2.txt. It is placed in Media. The LF represent linefeeds.



```
'The busy man has few idle visitors; to the boiling pot the flies come not.'LF
'If you would reap praise you must sow the seeds, Gentle words and useful deeds.'LF
'Anger is never without a reason but seldom with a good one.'LF
'Virtue and a trade are a child's best portion.'LF
'Love your neighbor Yet don't pull down your hedge.'LF
'He that does what he should not shall feel what he would not.'LF
'The honest man takes pains and then enjoys pleasures; The knave takes pleasures and then suffers pain:
'What you would seem to be, be really.'LF
'Necessity never made a good bargain.'LF
'The heart of a fool is in his mouth, but the mouth of a wise man is in his heart.'LF
```

There is a potential problem creating a file for use as a csv table using Windows based text editors. A text editor like *Notepad* creates both line feeds (LF) and carriage returns (CR) at the end of each line. Apple editor products only generate the LF. What App Inventor expects at the end of each line in the text table is a \n (a line feed). The line feed is needed to use the ***list from csv table*** block that feeds the LIST *thePhrases*. I use *Notepad++ text editor* to remove the CRs https://www.google.com/search?q=notepad%2B%2B&oq=notepad%2B%2B&aqs=chrome..69i57.5343j0j7&sourceid=chrome&es_sm=93&ie=UTF-8 and render the phrase2.txt text into UTF-8 format; not ascii. The example text in the graphic shows the CRs removed.

A simple alternative is to use a single line csv instead of the table, using only commas between the items. Then, use **list from csv row** block to convert the file to a list.

The Images

One of the benjaminfranklin images is a jpg file, the "speaking" image is a png image. The talking image was in-expertly edited, you probably can do better. :) Users do not need to have an image.

Translation / Language

A Listpicker object is chooses the translation language and country in a single step required by the TextToSpeech (TTS). When a language is chosen, the ListPicker changes the Country, Language and YantexLanguage variables simultaneously All of the translations are made from English to whatever languages are available in Yandex and TextToSpeech.

Translation is only available if your device has an active Internet connection. Whether the app can translate depends on the Yandex server being accessible.

If a “better” translation of your tips or phrases is required, use paired lists or another mechanism other than the Yandex Control to provide the translated text. The *Polyglot* tutorial might give developers ideas for additional “localization.”

What Languages does *Poor Richard* know? The app currently knows eight. The language and country codes the TTS and Yandex engines use in *Poor Richard* are:

TTS country: CZE,USA,DEU,FRA,NLD,ITA,POL,USA (csv used in Country variable in the app)

TTS language: ces,spa,deu,fra,hid,ita,pol,eng (csv used in Language variable in the app)

Yandex language: cs,es,de,fr,nl,it,pl,en (csv used in the YantexLanguage variable in the app).

The languages the *Poor Richard* ‘knows’ now and that are programmed are: Czech,Spanish,German,French,Dutch,Italian,Polish,English (csv used in ListPicker1.ElementsFromString in the app).

The language codes Yandex uses are different that what the TTS engine requires. All the codes are put into Lists are ordered so the first item in the TTS corresponds to the first TTS language and the first Yandex language.

Some language/country TTS codes are here: http://ai2.appinventor.mit.edu/reference/components/media.html#TextToSpeech ; Search the Internet for more, there are plenty more languages TTS knows. The following may work: http://www.science.co.il/language/locale-codes.asp .

Some of the Yandex language codes from http://voxcommando.com/forum/index.php?topic=1444.0 are:

Yandex Country Codes -- unofficial

ar =Arabic
az =Azerbaijani
be =Belarusian
	fr =French
	he =Hebrew
	hr =Croatian
		nl =Dutch
		no =Norwegian
		pl =Polish

bg =Bulgarian	hu =Hungarian	pt =Portuguese
bs =Bosnian	hy =Armenian	ro =Romanian
ca =Catalan	id =Indonesian	ru =Russian
cs =Czech	is =Icelandic	sk =Slovak
da =Danish	it =Italian	sl =Slovenian
de =German	ka =Georgian	sq =Albanian
el =Greek	lt =Lithuanian	sr =Serbian
en =English	lv =Latvian	sv =Swedish
es =Spanish	mk =Macedonian	tr =Turkish
et =Estonian	ms =Malay	uk =Ukrainian
fi =Finnish	mt =Maltese	vi =Vietnamese
		zh =Chinese

It was difficult to find a list of the current Yandex language codes. Enjoy the ones I was able to figure out above.

Yandex is the control used for translations, see the App Inventor documentation for more details.

TextToSpeech

Not all phones/tablets have a Text to Speech (TTS) engine available.   You may have to install a TTS from Google Play unless you have a newer Android phone or tablet.

Android TTS only has a female voice available on my phone. The voice is made more masculine by lowering the pitch value of the TTS. The lowest sounding pitch is a value of 0, the default is 1, and a very highest pitch is 2. Decimal values are permitted. The TextToSpeech.Pitch set with code in the app is a value of 0.3.

The speech rate is adjustable. You may want to slow the speaking rate down to understand what the TTS says; once again the range is 0 to 2 with the fastest speech at 2. In the app, set TextToSpeech.SpeechRate to 0.8 (the default is 1) and certainly try other settings as the rate can be set by the Slider.

To get proper inflections in various languages, you set TextToSpeech.Language and TextToSpeech.Country using the codes discussed earlier. The code in the AfterPicking event of the ListPicker sets TTS Language, TTS Country and also the Yandex language setting in one operation based on the Country selected from the options in the ListPicker.

To Do

- Provide your own translations (see the *Polyglot* tutorial) and eliminate *Yandex*.
- Add more languages that can be selected for both the TTS and Yandex.
- Experiment with the speech and pitch rates.
- Add options to save favorite phrases using a TinyDB for future user enjoyment.
- Add options to never repeat a selection (remove items from *thePhrases* list as the sayings are used perhaps) - especially if this will be a Tip of the Day app with 366 or so items.
- Add an option to repeat the previous saying. The published aia file does that now. The blocks are not presented in the tutorial but are included in the aia Allowing the repeat of a current selection required two variables, the addition of a button (Button4) and a small code modification in Button1. Review the aia to see how this was accomplished with only a few lines of code.
- Add ability to turn off both TTS and Yandex or build the app without these controls.
- Add capability to check if there is an Internet connection. If there is not one, turn off the translate functions and hide or gray out (disable) the translate option buttons.
- Make the Speech rate settings persistent (save to a TinyDB and read the value in the Screen1.Initialize block when the app starts) or any other setting.

Is there a source file? **Download Source Code**

Download the source code to your computer, then open App Inventor 2, click Projects, choose Import project (.aia) from my computer..., and select the source code you just downloaded. There is a single aia file: **PoorRichard.aia** (the complete project). The link to the Google folder where the aia is stored is on the *App Inventor for Fun Poor Richard* tutorial blog page.