Introducing Keyman

Keyman is created by SIL International

SIL @ NPIC

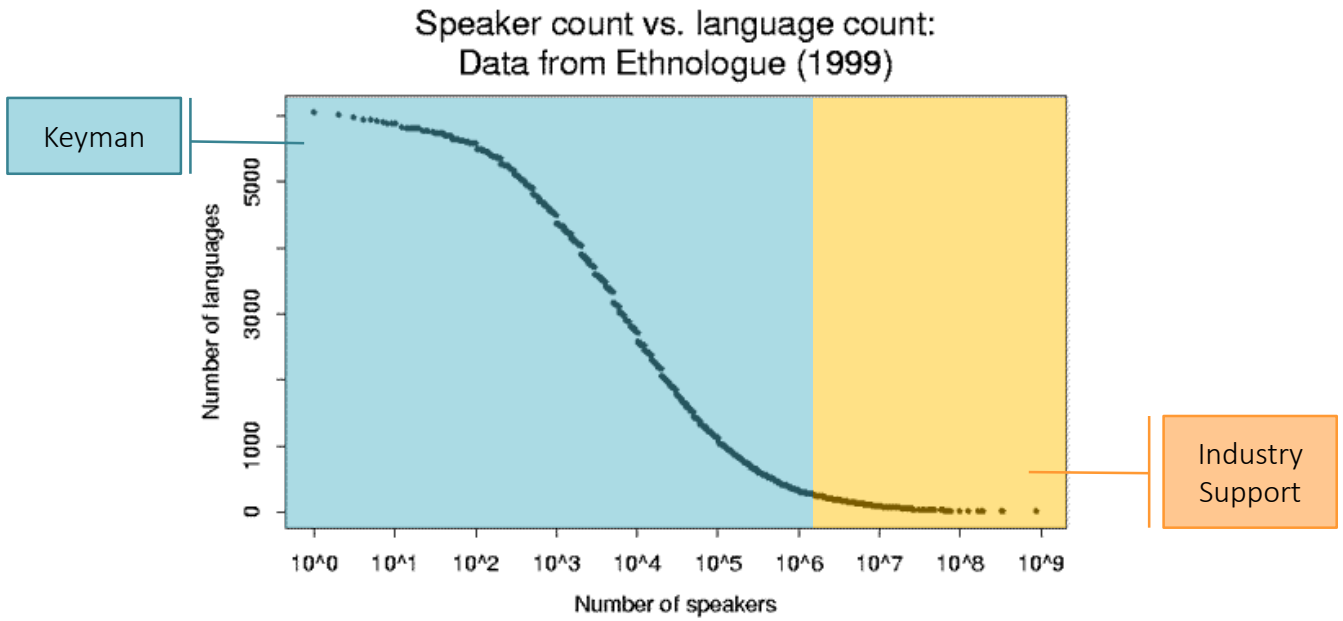
- The SIL Language Software Development Unit at NPIC started work in May 2017
- Our first project at NPIC will be Keyman

What is Keyman?

- The Keyman project is a set of Open Source tools for creating and using keyboard layouts for any language in the world

Why does Keyman exist?

- Industry supports about 200 languages
  - Top priority on Latin script
- **But!** 7,099 world languages
  - Cambodia – 19 languages
  - Thailand – 50 languages
  - Laos – 71 languages
  - Myanmar – 112 languages
  - Papua New Guinea – 839 languages!
- Keyman supports the “long tail” of languages



Data from ethnologue.com

Why use Keyman?

- Keyman was born in South-East Asia
- Keyman emphasises equal support for all languages:
  - South-East Asian languages
  - Indic languages
  - Right-to-left scripts such as Arabic
  - Minority languages

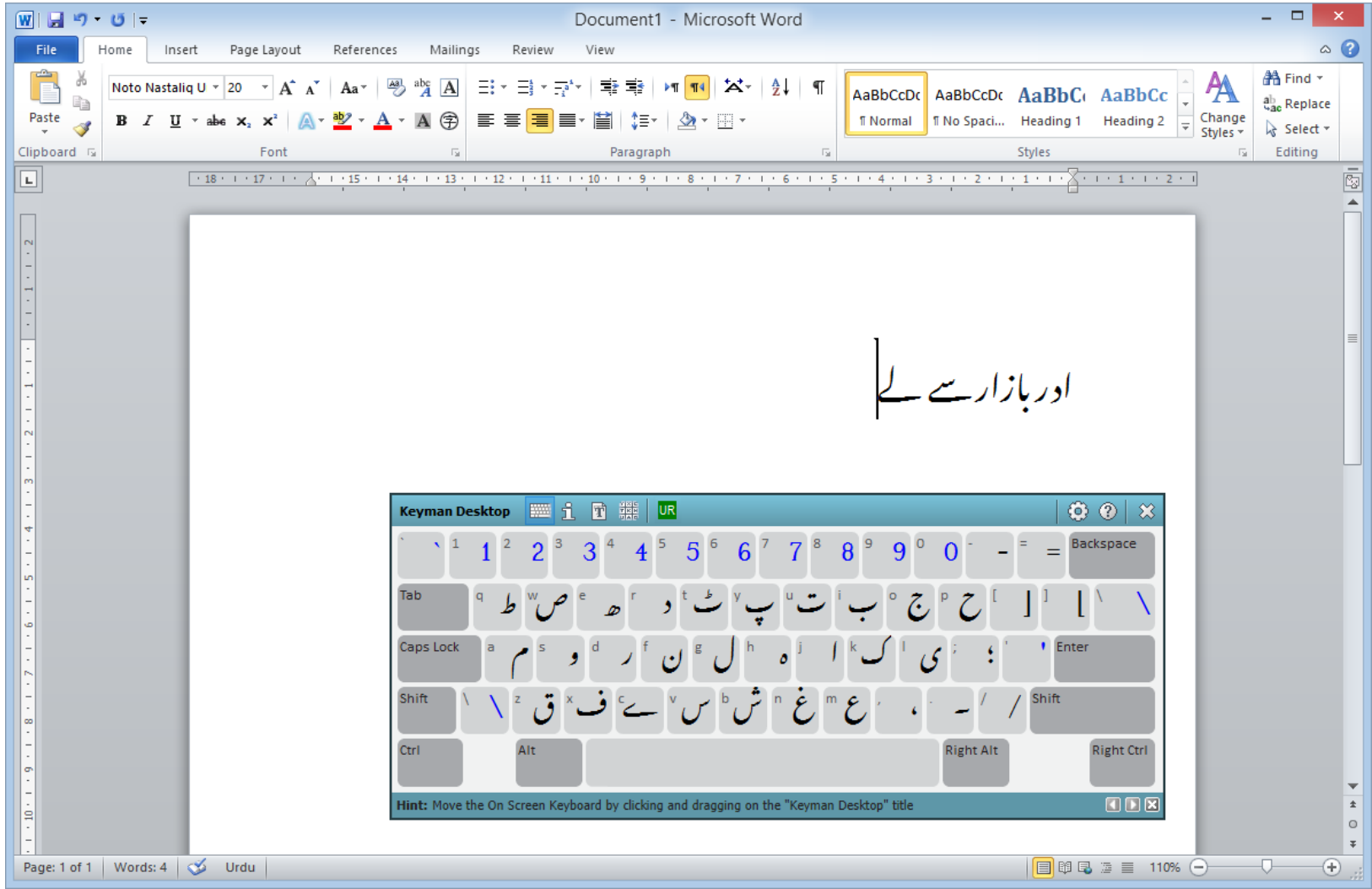
Who uses Keyman?

- SIL members worldwide and associates – many thousands
- Large multilingual organisations, e.g. BBC World Service
- Governments, e.g. Indian state governments, Canada, Australia
- Non Government Organisations – many around the world
- Individuals and businesses – thousands more each week

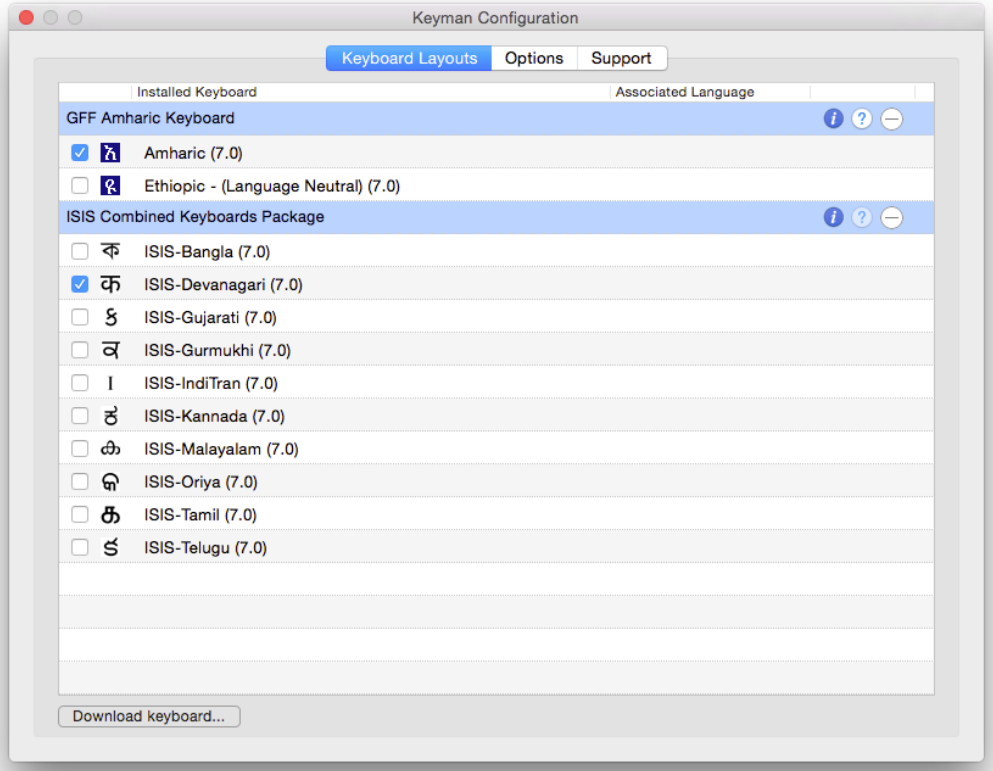
Keyman works on many systems

- Windows
- Mac OS X
- Android
- iPhone and iPad
- Web
- Linux (with the Keyman KMFL clone)

Keyman runs on Windows



Keyman runs on Mac OS X

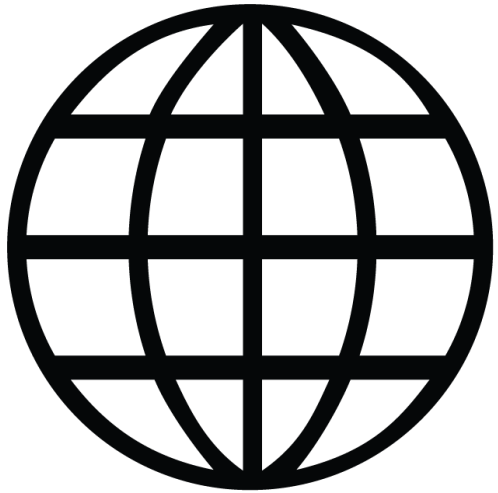


Keyman runs on iPhone and iPad

Keyman runs on Android

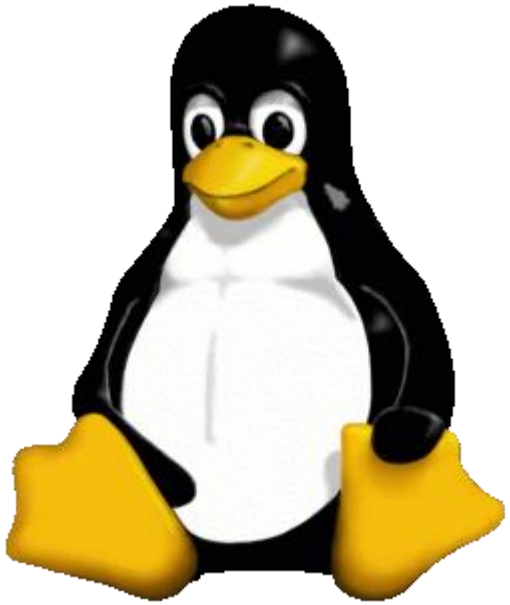
Keyman runs on the Web

- Desktop ***and*** mobile
- www.keymanweb.com



Keyman even runs on Linux

- A Keyman clone called KMFL is available for Linux
- kmfl.sourceforge.net



A Brief History of Keyman

- 1992 – Keyman 1.0
  - For Windows 3.1
  - Lao language
  - Not Unicode

- 1994 – Keyman 3.0
  - Still for Windows 3.x
  - Supported user-defined keyboard layouts
  - Used for many languages in South East Asia and elsewhere
- 1996 – SILKey
  - Clone of Keyman released for Mac OS 9

A Brief History of Keyman

- 2000 – Keyman 5.0 
  - Unicode support

- 2002 – Keyman 6.0 
  - Windows XP means Unicode starts to become better used

- 2006 – KeymanWeb 1.0 
  - Keyman keyboards run as JavaScript on WWW

A Brief History of Keyman

- 2013 – Keyman for iPhone and iPad
- 2014 – Keyman for Android
- 2014 – KeymanWeb 2.0 is Open Source
  - Supports desktop and mobile browsers
- 2014 – Keyman Desktop 9.0 
  - Deep support for Windows 8 and later versions
- 2015 – Keyman for Mac OS X
  - First beta release
- **2015 – SIL acquires Keyman**

Why use Keyman?

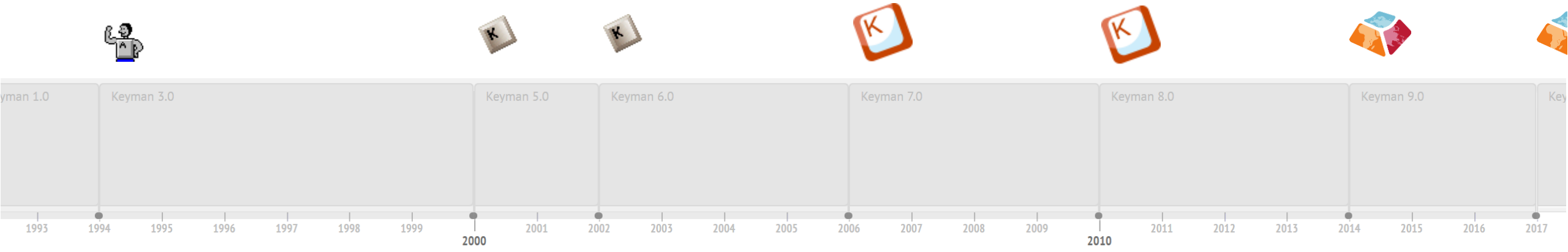
- Keyman advantages
  - Open source
  - Robust and well supported
  - Library of existing keyboard layouts
  - On-screen and hardware keyboards
- Unique features
  - Cross-platform
  - Development tools
  - Powerful and flexible keyboard layouts

Advantage: Open Source

- The Keyman project is going completely Open Source
- This will be our initial goal at NPIC
- https://github.com/keymanapp

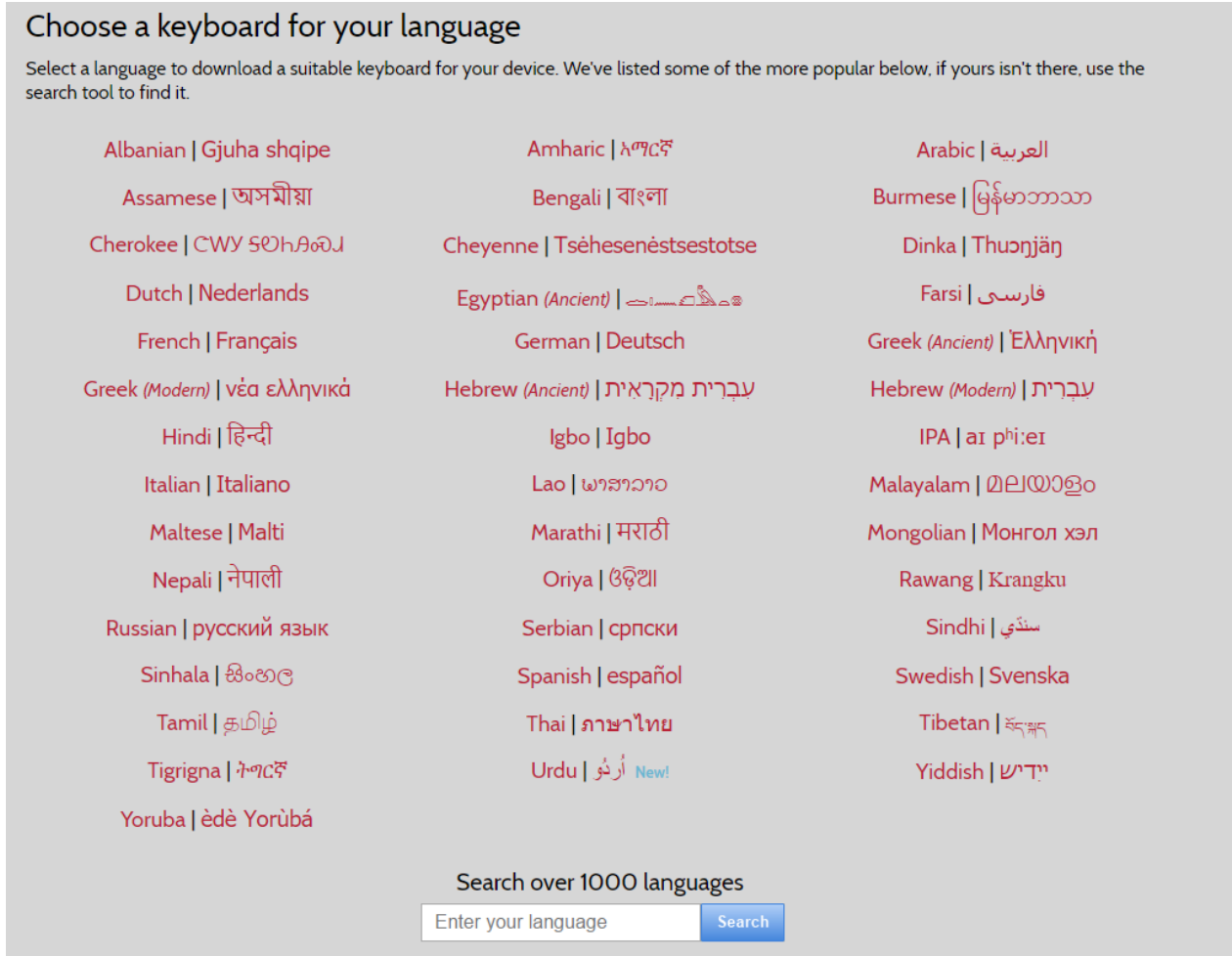
Advantage: Robust and well supported

- Keyman Desktop has been under active development for 25 years
- Keyman-specific support in many products, e.g. LibreOffice, Firefox
- Microsoft have even added features in Windows to support Keyman



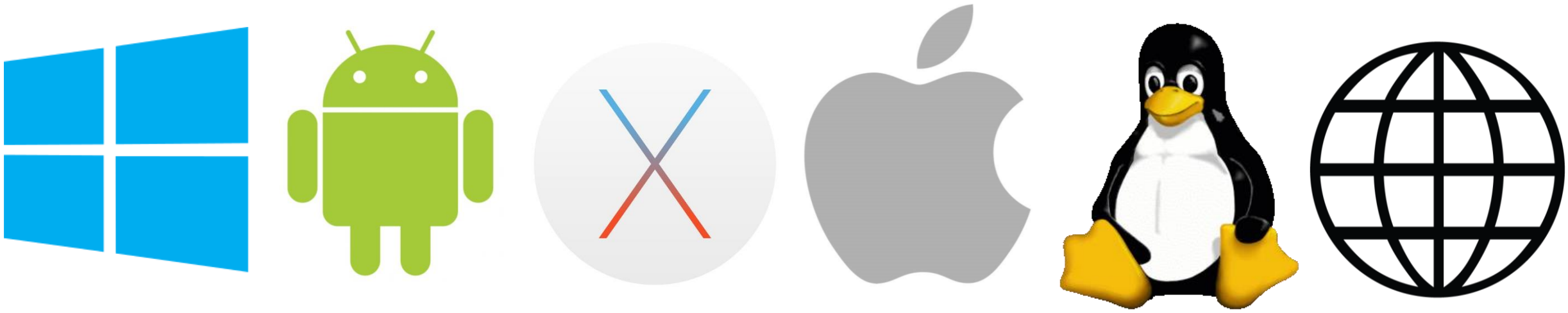
Advantage: Library of keyboard layouts

- Over 1,000 different languages
- Many languages have multiple layouts
- Future goal: Full source code for all layouts



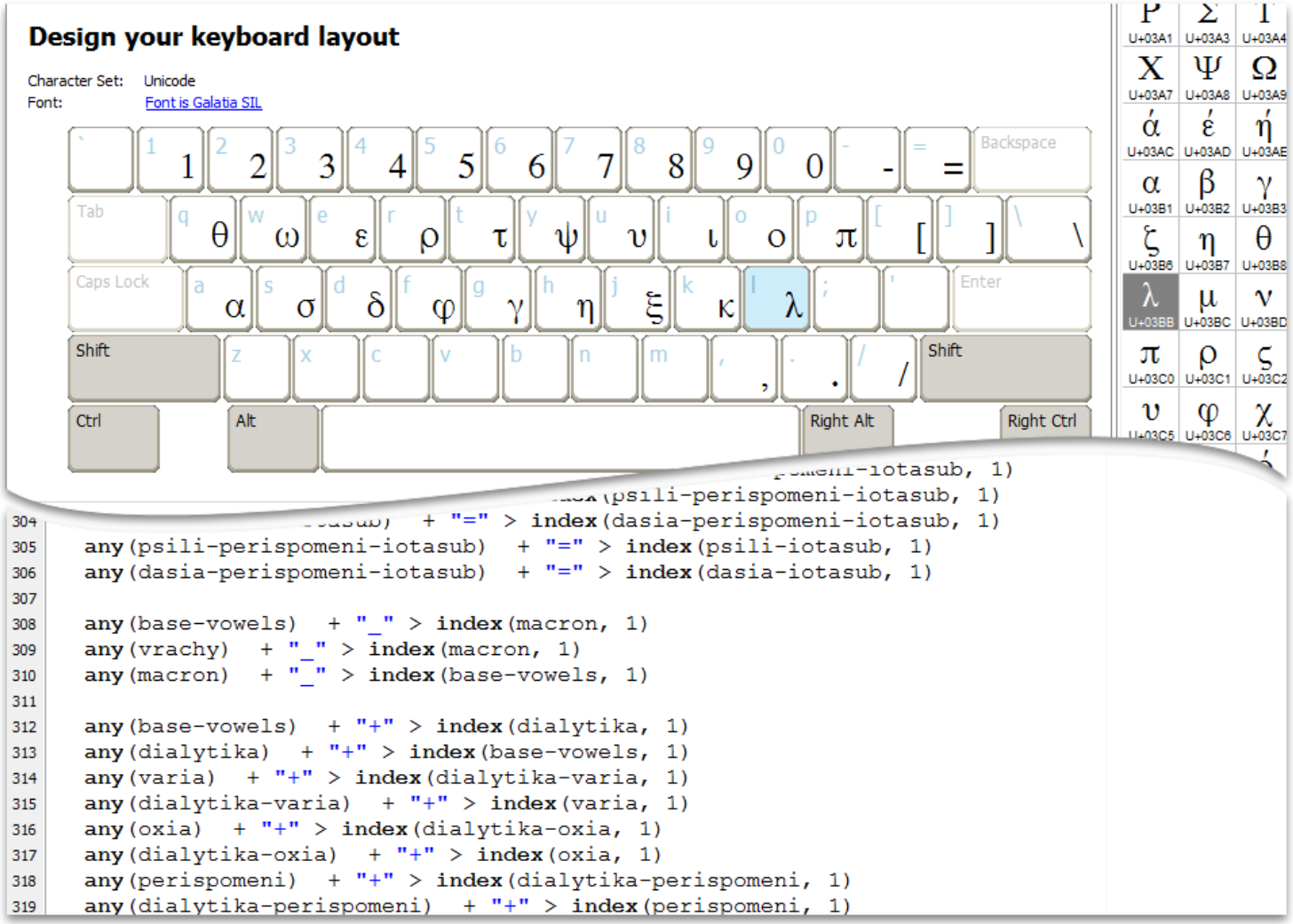
Feature: Cross-platform

- Every platform has a different input method system
- There are no other established open multi-platform input method systems
- Keyman provides a common base system for input methods
- Develop a keyboard layout once, use it everywhere



Feature: Development Tools

- Keyman has powerful tools to create keyboard layouts
- Visual and code-based tools
- We will introduce Keyman Developer in the next session



Vision: LDML + Unicode

- SIL have been asked by the Unicode Consortium to develop an open standard for keyboard layouts
- Keyman will be the reference platform for this open standard
- Brings the opportunity to be visible on the global stage as the standard is finalised and implemented by major industry vendors
- More about LDML and Unicode on Thursday

Developing Keyboards

A Keyman Developer Tutorial

- An introduction to creating Keyman keyboards based on Khmer NiDA keyboard

Discussion Questions

- Is it easy to learn to type Khmer with Unicode?

- Can you remember learning to type?
- What sorts of problems did you experience?
- Did you learn to type on Khmer Limon keyboard?
- Was it easy to move to Khmer NiDA?

Our first word of the day

Encoding “ខ្មែរ”

- First, understand the encoding
- ខ្មែរ in Unicode is encoded as:

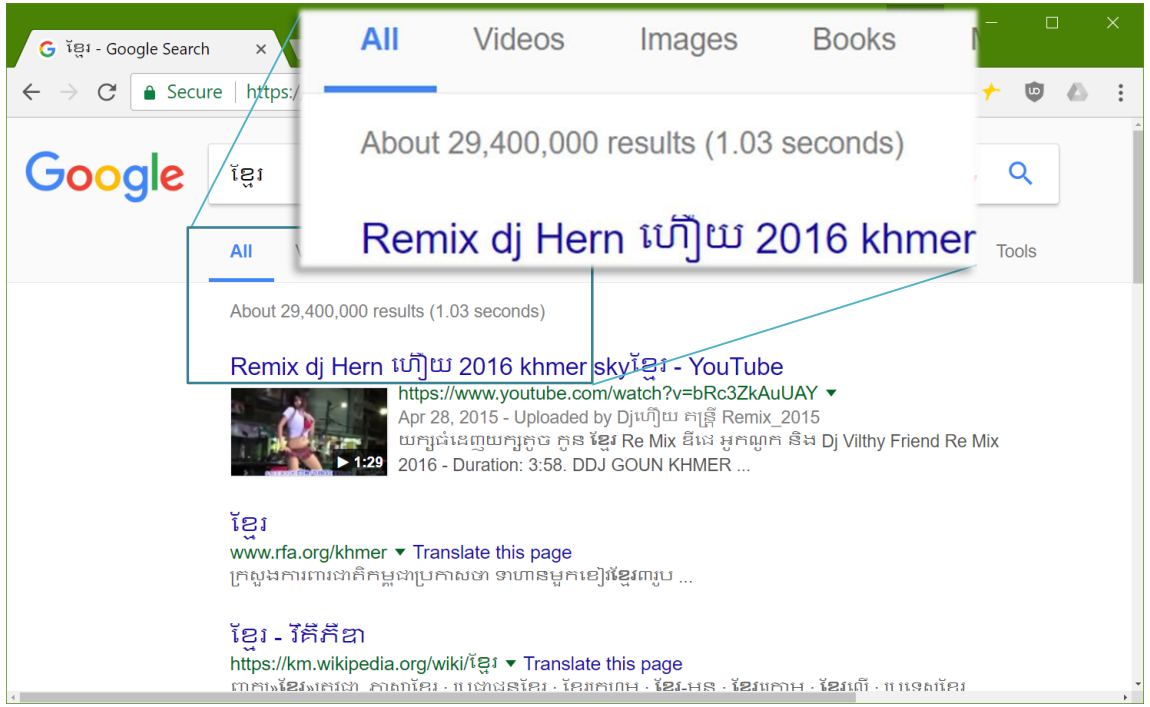
ខ	◌្	ម	◌ែ	រ
U+1781	U+17D2	U+1798	U+17C2	U+179A

- or as:

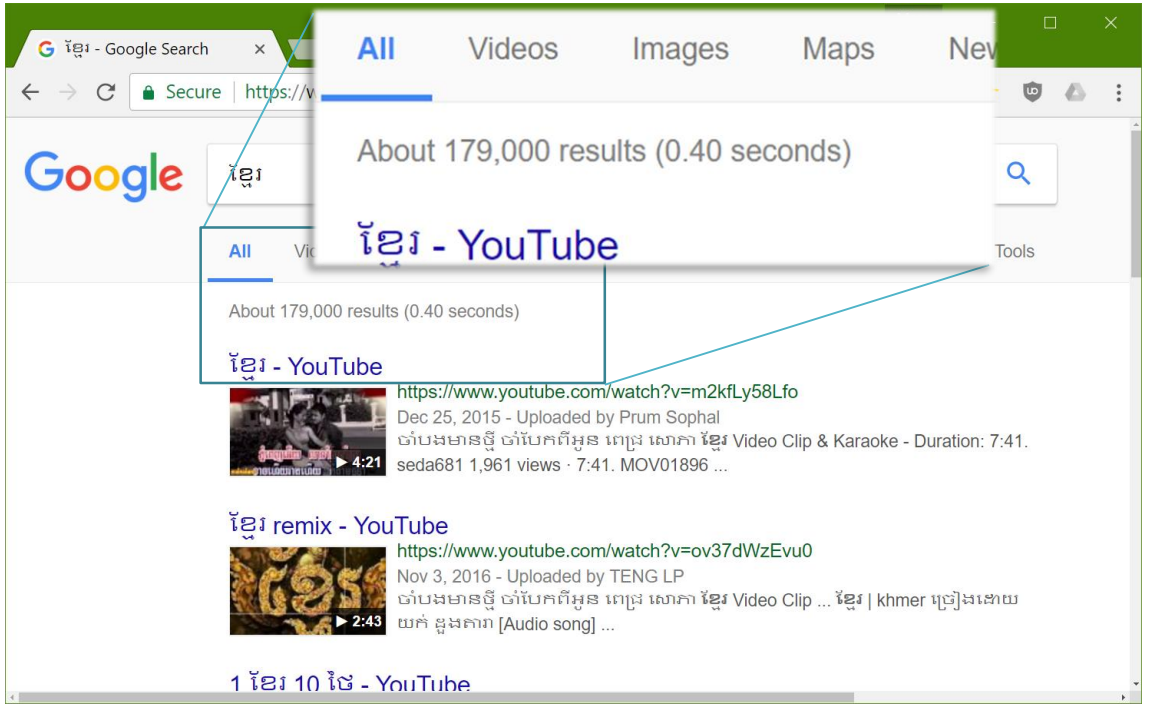
ខ	◌ែ	◌្	ម	រ
U+1781	U+17C2	U+17D2	U+1798	U+179A

Google Fight: ខ្មែរ vs ខ្មែរ

ខ	◌្	ម	◌ែ	រ
U+1781	U+17D2	U+1798	U+17C2	U+179A



ខ	◌ែ	◌្	ម	រ
U+1781	U+17C2	U+17D2	U+1798	U+179A



Typing to match encoding

ខ	◌្	ម	◌ែ	រ
U+1781	U+17D2	U+1798	U+17C2	U+179A
X	J	M	SHIFT E	R

ខ	◌ែ	◌្	ម	រ
U+1781	U+17C2	U+17D2	U+1798	U+179A
X	SHIFT E	J	M	R

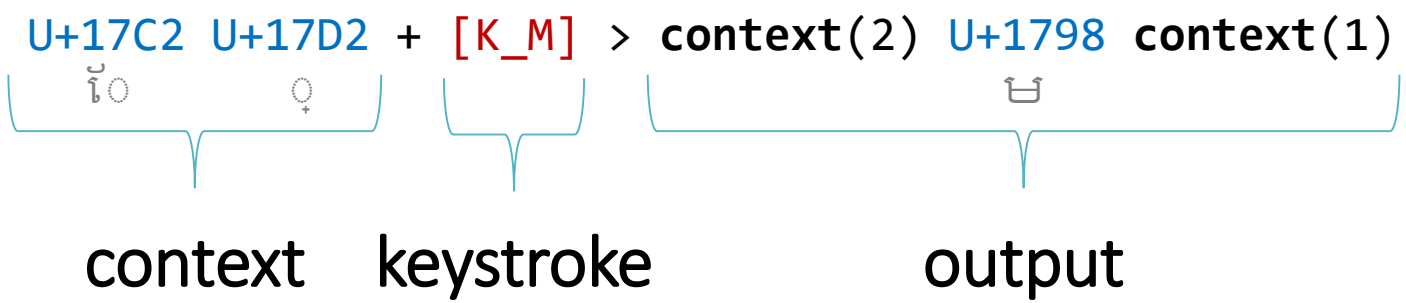
Keyman Code to support “ខ្ញុំ”

```
c This keyboard definition supports only one word: ខ្ញុំ

store(&VERSION) '9.0'
store(&NAME) 'Basic Khmer'
begin Unicode > use(main)

group(main) using keys

c Keys for ខ្ញុំ

+ [K_X] > U+1781
+ [K_J] > U+17D2
+ [K_M] > U+1798
+ [SHIFT K_E] > U+17C2
+ [K_R] > U+179A
```

Add a rule to reorder the user’s input

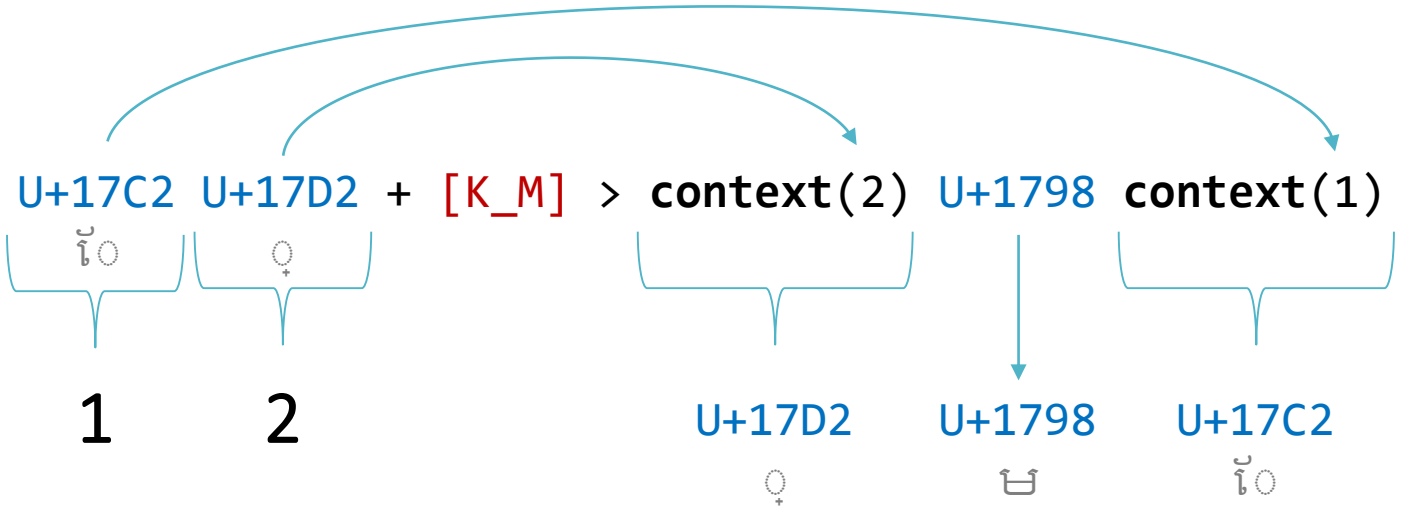
```
c Keys for ខ្មែរ

+ [K_X] > U+1781 ខ
+ [K_J] > U+17D2 ្
+ [K_M] > U+1798 ម
+ [SHIFT K_E] > U+17C2 ែ
+ [K_R] > U+179A រ

c Fixup when user types vowel before coeng+m

U+17C2 U+17D2 + [K_M] > context(2) U+1798 context(1)
```

Anatomy of a Keyman keyboard rule



- **context**: what is already on screen (“text store”)
- **keystroke**: what the user types
- **output**: a transform which replaces *context* in the text store.

Anatomy of a Keyman keyboard rule



- **context()** statement copies a character from the *context* to the *output*

Introducing our next word

Encoding “ស៊ី”

- ស៊ី in Unicode is encoded as:

ស	◌៊	◌ី
U+179F	U+17CA	U+17B8

- but people sometimes type:

ស	◌ុ	◌ី
U+179F	U+17BB	U+17B8

ស	◌ី	◌ុ
U+179F	U+17B8	U+17BB

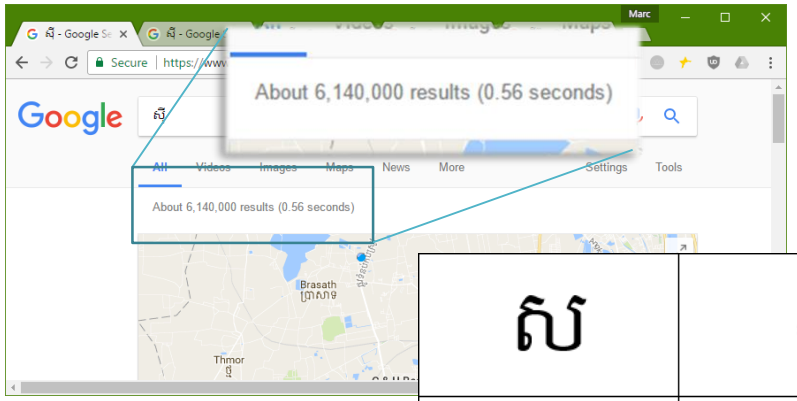
ស	◌ិ	◌ី
U+179F	U+17C9	U+17B8

ស	◌ី	◌៊
U+179F	U+17B8	U+17CA

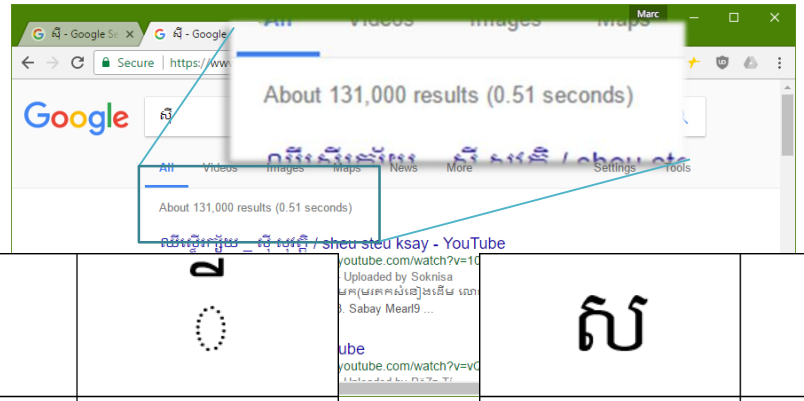
I love iPhone!

Google Fight: ស៊ី vs ស៊ី vs …

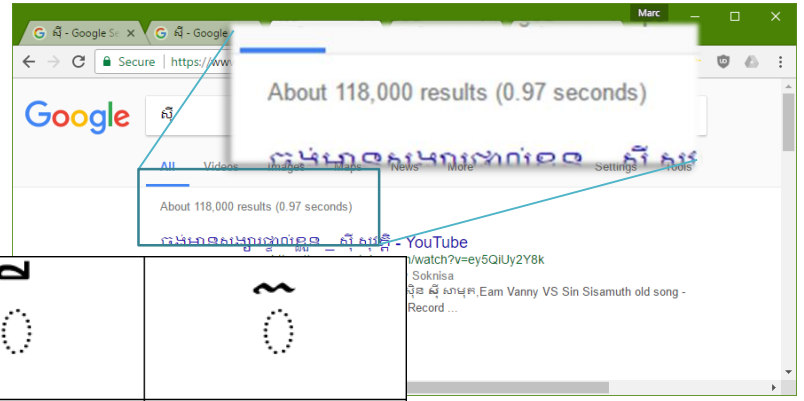
ស	◌ៈ	◌ី
U+179F	U+17CA	U+17B8



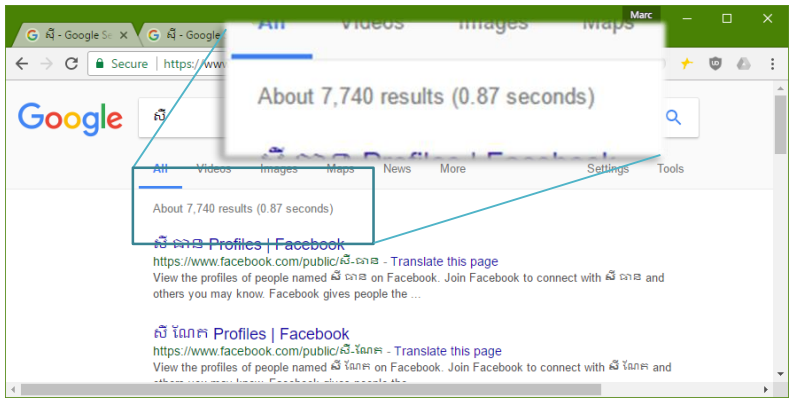
ស	◌ុ	◌ី
U+179F	U+17BB	U+17B8



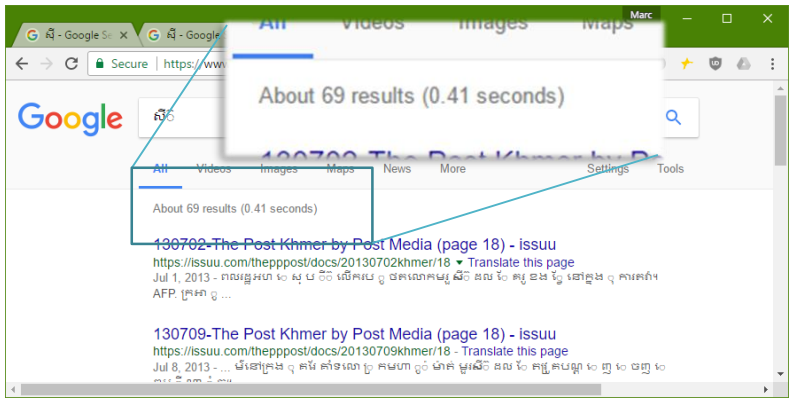
ស	◌ី	◌ុ
U+179F	U+17B8	U+17BB



ស	◌៉	◌ី
U+179F	U+17C9	U+17B8



ស	◌ី	◌ៈ
U+179F	U+17B8	U+17CA



Typing “ស្ទី”

ស	◌៊	◌ី
U+179F	U+17CA	U+17B8

ស	◌ុ	◌ី
U+179F	U+17BB	U+17B8

ស	◌ី	◌ុ
U+179F	U+17B8	U+17BB

ស៊ី Keyman Code

```
c This keyboard definition supports only one word: ស៊ី

store(&VERSION) '9.0'
store(&NAME) 'Basic Khmer'
begin Unicode > use(main)

group(main) using keys

c Keys for ស៊ី

+ [K_S]       > U+179F   c ស
+ [K_U]       > U+17BB   c ុ
+ [K_SLASH]   > U+17CA   c ៊
+ [SHIFT K_I] > U+17B8   c ី
```

Correcting the output

```
c Keys for ស៊ី

+ [K_S]       > U+179F   c ស
+ [K_U]       > U+17BB   c ុ
+ [K_SLASH]   > U+17CA   c ៊
+ [SHIFT K_I] > U+17B8   c ី

c Fixup when user types in the wrong letter

U+179F U+17BB + [SHIFT K_I] > context(1) U+17CA U+17B8
U+179F U+17B8 + [K_U] > context(1) U+17CA context(2)
```

Correcting the output – analysis

```
U+179F U+17BB + [SHIFT K_I] > context(1) U+17CA U+17B8
```

ស ◌ុ ◌៊ ◌ី

```
U+179F U+17B8 + [K_U] > context(1) U+17CA context(2)
```

ស ◌ី ◌៊

- Can you suggest a rule to correct this mistake?

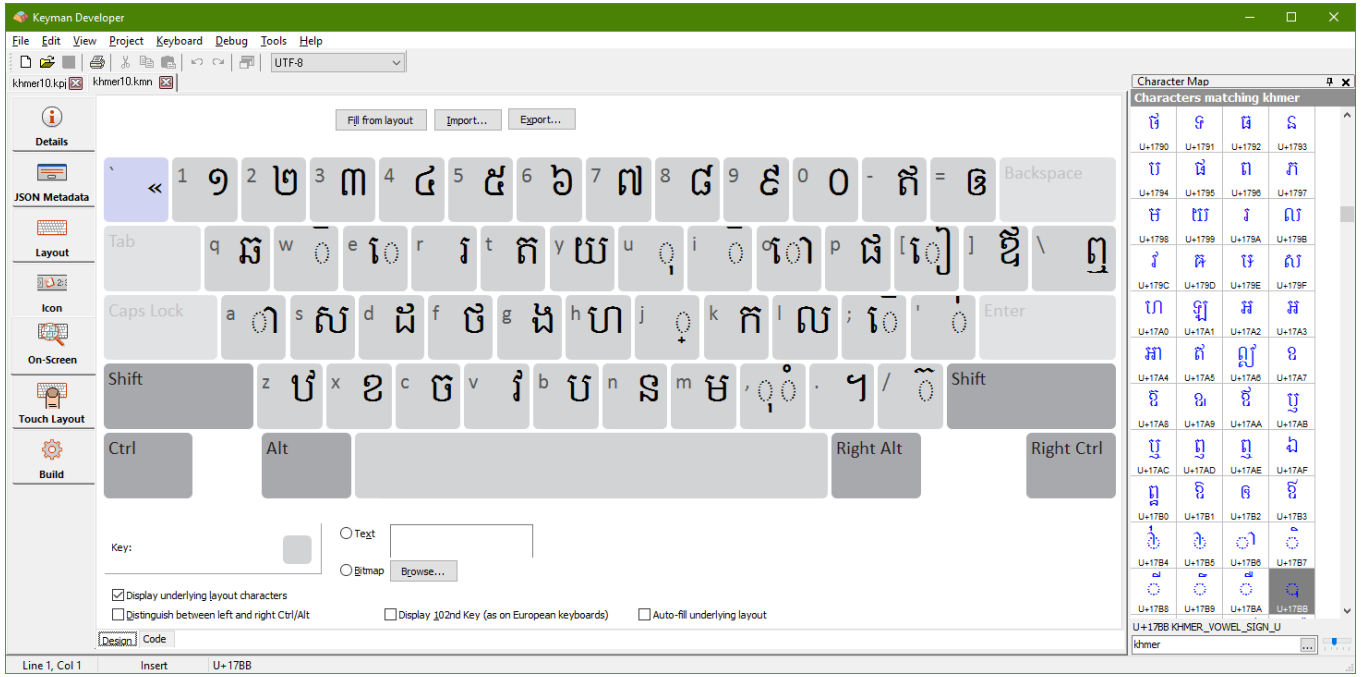
ស	◌៊	◌ី
U+179F	U+17C9	U+17B8

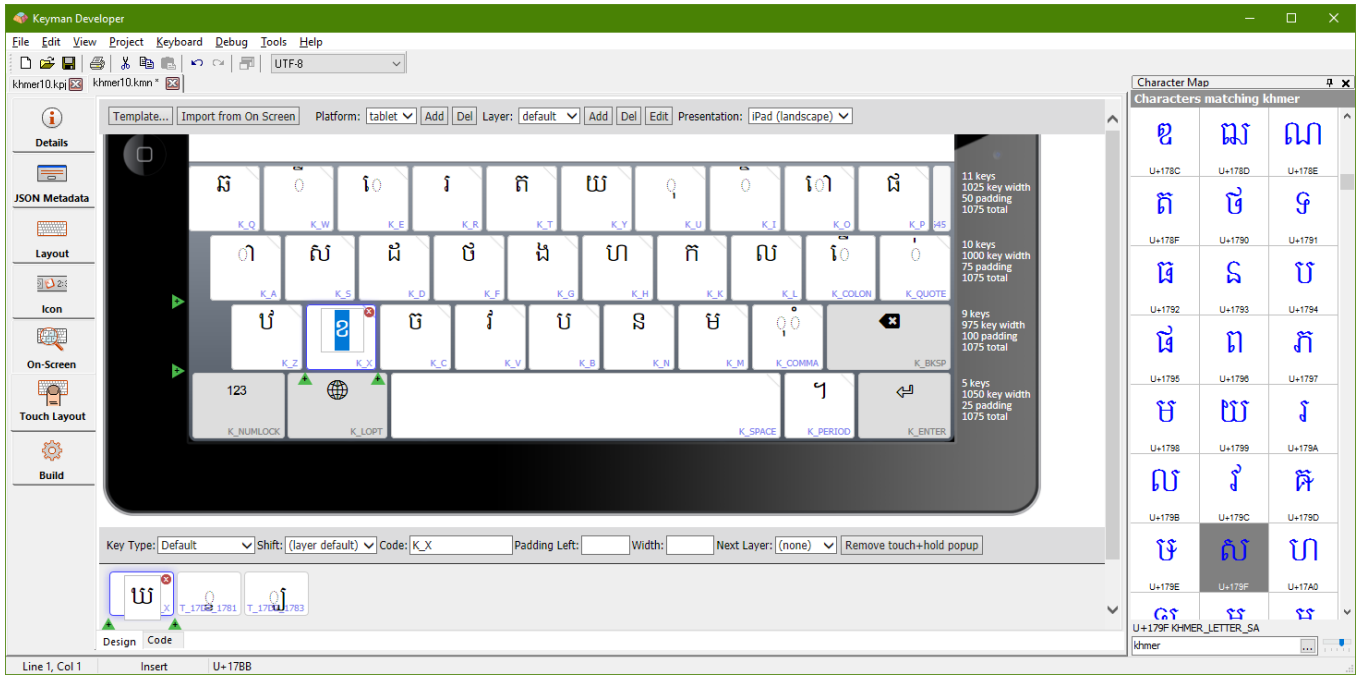
Keyboard Rules - Summary

- **Reorder** to fix mistyped words - ខ្ញែ
- **Block** invalid keystrokes - ស្ិ + ◌ី
- **Substitute** incorrect letters - ស្ិ
- **Combine** decomposed letters - ◌េ + ◌ី → ◌ើ
- **Break Apart** composed letters

Multiplatform Keyboards

- Keyman Developer rules apply on every platform
- But presentation of the keyboard can vary
- Import basic layout from Desktop to Touch
- Optimise for use on small devices





Deploying Keyboards

- Create a keyboard layout in Keyman Developer
- Push to GitHub
- Create a Pull Request on **https://github.com/keymanapp/keyboards.git**
- When accepted, the layout is automatically available on every device through Keyman keyboards menu
- Full details at **https://help.keyman.com/developer/keyboards**

Question and Discussion Time

- Learn more:
  - www.keyman.com
  - www.sil.org