Automating **Color-coding** for

Pronunciation-Guided Tibetan Text

Using regular expressions to generate HTML color codes for the four main sound profiles within central standard Tibetan.
Introduction

Reading is a complex and difficult skill. The main difficulty beginning readers face is learning which letters represent which sounds—and then getting used to those patterns by reading them, again and again, in different combinations and contexts. It takes practice to learn how to read. Research also shows that the easier reading is, the more likely it is that readers will read. Readers are discouraged by difficult texts; they are encouraged by easier ones.

That’s why the goal of the author, editor, and publisher of today is to create easy-to-read text. When we think about what makes a text easy to read, we first think of the content. Are the words the author uses simple and clear? Or is the vocabulary unnecessarily complex? Do the sentences proceed with pace and punctuation? Or do they weave a dense, tangled web that the reader struggles to parse?

Just as important as clear content, however, is clear presentation. Font, font size, punctuation, spacing—these all have a measurable impact on ease-of-reading. Text that’s too small is hard to read; yet it can’t be so large that the reader can’t “see where they’re going.” Punctuation provides prosody cues—pauses that help readers chunk together meaning. And spacing keeps words from running onto one another. The basic idea is this: The more work the text does, the less the reader has to do.

If we’re thinking about beginning readers, it’s important to maximize the work we can do for them. The text needs to do as much of the heavy lifting as possible. In the case of Tibetan language learning materials, the question is then this: How can we provide beginning readers with the maximum amount of clear, consistent clues that link symbol to sound? The answer, I think, is something like this:

1. At the level of the sentence, we can **add spaces or punctuation** between words—word-spacing is work the text can do so that beginning readers aren’t confused about “what’s a word” within a continuous string of text
2. We can **add punctuation** to give prosody cues—cues about where to pause and how to inflect—so that the rhythm reflects the meaning
3. And, at the level of the word, we can **add color-coding** to make it clear how the word is to be pronounced

This last point is the focus of this paper. In “transparent” or “shallow” writing systems, spelling-to-pronunciation is easy, simple, and clear. (And, research tells us that children learn to read such languages faster). When the Tibetan writing system was invented, the spelling accurately encoded the language’s pronunciation. Due to 1,500 years of language change, however, Modern Tibetan has become a “deep orthography”—that is, the relationship between symbol to sound is not 1:1.
And yet the pronunciation has shifted in predictable patterns. The goal of our color-coded text, then, is to make these patterns easy for beginning readers to see...

The Four Sound Profiles of Tibetan

Tibetan’s alphabet is organized in 4 columns, and sounds shift through these columns in a predictable manner. The aspects of phonology that are important here are 1) aspiration and 2) tone (or accentuation). Differences in these sounds distinguish meaning in Tibetan—they also tend to be distinctions not used in Western languages. This also makes them especially important for second-language learners. For convenience’s sake, let’s simply take a look at how the first row of the alphabet behaves with regard to aspiration and tone (ka, kha, ga, nga):

<table>
<thead>
<tr>
<th>unaspirated</th>
<th>aspirated</th>
<th>aspirated</th>
<th>unaspirated</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
</tr>
</tbody>
</table>

The four sound profiles are:

- **Unaspirated, high sounds**: Column 1 sounds are **unaspirated**, meaning breath isn’t expelled when speaking them—in English, for example, you can hear a difference between the “p” sound in “pin” (aspirated) and “spin” (unaspirated)—and **high** (higher in pitch, **sharper** in accentuation, and **shorter** in duration)
- **Aspirated, high sounds**: Column 2 sounds are **aspirated** (breath is expelled) and **high**
- **Aspirated, low sounds**: Column 3 sounds are **aspirated** and **low** (lower in pitch, **duller** in accentuation, and **longer** in duration)
- **Unaspirated, low sounds**

We’re able to represent these four sound profiles by giving them each a unique color. Here, yellow, red, blue, and green. Another reason we’ve chosen these distinctions in particular—aspiration and tone—is that these principles carry over into pronunciation at the syllable level. As we mention above, modern Tibetan is a “deep” orthography. When it comes to decoding spelling, it is extremely helpful to recognize that, while not all letters are pronounced, many of them affect the aspiration or tone! For example:
When an unaspirated, low sound (such as *yag*) is spelled with a prefix or head letter, it changes in tone (*g.yag*, "yak", high tone); an aspirated, low sound, on the other hand (*bu*), loses its aspiration (*'bu*). Color-coded text can thus work as a context clue for beginning readers when it comes to how to pronounce a certain word. Again, the more work the text does, the less the reader has to do.

However, for the text to do more work for the reader, the author must do more work on the text! Traditional writing methods would require a pen-and-ink switch for each letter we wrote. Logistically speaking, traditional word editors require just as much work. We would have to manually highlight each and every syllable of text, and select the color again and again. Fortunately, machines provide us with a solution for automatically generating color codes for Tibetan input...

### Automating Color-Coding

Our input in this case is plain text—UTF 8 Tibetan unicode. Stored as plain text, files can be given a .XHTML extension, converted to color, and easily added to an EPUB directory. The conversion is simply a process that performs a regular expression replacement. The regular expression itself is a pattern-matcher. It searches for a string of text fitting a certain pattern; when it finds that string of text, it then replaces it with (in this case) the text itself plus HTML <span> tags that define the text’s color.

### Using Regex to Recognize Tibetan Syllables

Let’s take a closer look at the first regex (regular expression) as an example. Here is the regex code for our first-column letters. It will recognize any Tibetan syllable where the main letter is first column, without a prefix or headletter (the yellow sound profile; unaspirated, high tone):

```regex
(unicodeLiteral|\d+(\s+)?[a-fA-F]+)(\d+(\s+)?[a-fA-F]+){1}(\d+(\s+)?[a-fA-F]+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)?\d+){1}(\d+(\s+)
The general pattern here is:

\((r'ABC', r'XYZ')\)

Where ‘ABC’ is the pattern to be matched, and ‘XYZ’ is the replacement. Setting the replacement aside for the moment, let’s look at the regex pattern that we are looking for:

\(?<=[^\p{L}\s>])([^\p{L}\s>]+)?\)

We are asking the computer to search for a string of text. The first thing we do is a positive lookbehind: \(?<=[^\p{L}\s>])\). That is, find a string of text whose first character is following one of these characters: a piece of Tibetan punctuation (a *tsheg* or a *shad* or a double *tsheg* or a Tibetan parentheses) or any type of space character (\(\s\)).

Since we are generating our own input, I know what to expect behind each Tibetan syllable. For a broader application, more characters may need to be added to this list; alternatively, one could also use a negative lookbehind and exclude the class of all Tibetan characters (in other words, just behind any valid Tibetan syllable will be anything but a Tibetan character).

Next, let’s search for the (un-prefixed) main syllable stacks for our yellow sound profile:
The yellow sound profile is exactly one (\{1\}) first-column consonant (ka, ta, pa, etc.) or (\{\}) a first-column consonant with a root letter. Here, I've only bothered to add codes for combinations that actually exist in standard Tibetan. I've also used the literal unicode code (\u0FB1, for example), as stacked characters listed one after the other tend to stack on top of one another, making the code unreadable.

I've also added a “capture group” by putting the whole list of possible yellow profile main stacks (this or this or this or this, a list separating each element with the regex symbol for “or”, the pipe: | ) in a set of parentheses. This group capture will be key for the replacement step, as it allows us to color-code the main stack as a bolder color than the prefix/postfix letters of the syllable! In other words, our code can recognize which group of letters is the main stack of the Tibetan syllable.

Next, we allow for an optional vowel by listing the four possible vowels' unicode values and following with a question mark (?) to indicate 0 or 1 vowels as a possible match for a legal syllable:

We do the same for the set of potential suffix letters:

\footnote{Here, we have the expectation of properly spelled input. Otherwise, it would be good to make a further separation between which letters are allowed to be followed by a post-suffix “sa.”}
Finally, we end it all with punctuation or a space character:

The following codes for the other sound profiles are all built around this same structure. Importantly, we run our codes for non-prefixied syllables first so we don’t end up with mistakes for main syllable stacks our regex can’t recognize automatically (ones that lack any vowel or headletter or rootletter).

By default, we know that well over 99% of legal Tibetan words without these markers either begin with the main letter, or have other illegal combinations that wouldn’t allow them to be mis-coded. (If we ran prefix possibilities first, we might mis-code words that begin those prefix letters as a main letter—for example, “dang” [“and”] as “dnga”—“nga” with a prefixed “d”. Catching “dang” first avoids these sorts of issues).

Replacing Plain Text with HTML Color Codes

Now, let’s take a closer look at our color-code replacement:

Above, we are simply using HTML <span> tags to indicate the RGB value of our colors. Remember that earlier, we made sure to surround each successive element of the Tibetan syllable with a set of parentheses (()). These performed a function called “capture group”—all these groups have been “captured,” meaning we can call them back with a simple code in the style of: \1 (to print the first group that matched our regex).

Thus, “\1\2\3\4\5” would simply re-print any Tibetan syllable matching the code. By placing the groups within span tags for different colors, we can modify which parts of the syllable are rendered in which colors. Here, we have given the main syllable stack a darker, bolder color
and the other parts of the syllable a lighter color. This will help beginning readers hone in on the main letter that is to be pronounced with accuracy and ease, for example:

Summary

Sample of Tibetan text run thru the color-coder...

A sample dialog rendered in the color-coding scheme...

By running our Tibetan text through our color-coder, we now have text that does even more work for the beginning reader. Our text gives the reader lots of extra information: Its color
tells the reader how it should be pronounced (in regards to tone and aspiration); and its bolding
tells the reader which letter is the **main letter** of the complex syllable stack. Meanwhile,
re-introducing the double *tshug* () gives the reader a cue as to which syllables come together to
form a word.

Despite all the work the text does for the reader, creating texts that do more work doesn’t
have to mean more work for authors; most of the heavy lifting can be outsourced to machines
by automating simple, repetitive tasks. In this case, we have given the machine the task of
inserting color-codes, letter-by-letter. The output is a highly readable pronunciation-guided
Tibetan text for beginning readers.

If we’re to give beginning readers of the Tibetan language the best chance for attaining
gradually more sophisticated levels of reading skill, it seems that making easy-to-read texts is
one of the most important contributions writers, authors, and book-makers can contribute to the
cause...
Appendix 1

Full list of regex codes: