Easysync Protocol

AppJet, Inc., with modifications by the Etherpad Foundation

March 27, 2011

1 Attributes

An "attribute" is a (key,value) pair such as `(author,abc123)` or `(bold,true)`. Sometimes an attribute is treated as an instruction to add that attribute, in which case an empty value means to remove it. So `(bold,)` removes the "bold" attribute. Attributes are interned and given numeric IDs, so the number "`6`" could represent "`(bold,true)`", for example. This mapping is stored in an attribute pool which may be shared by multiple changesets.

Entries in the pool must be unique, so that attributes can be compared by their IDs. Attribute names cannot contain commas.

A changeset looks something like the following:

```
Z:5g>1|5=2p=v*4*5+1$x
```

With the corresponding pool containing these entries (among others):

`4` → `(author,1059348573)`

`5` → `(bold,true)`

This changeset, together with the attribute pool, represents inserting a bold letter "x" into the middle of a line.

The string consists of:

- a letter `Z` (the "magic character" and format version identifier)
- a series punctuation marks (operation codes or "opcodes" for short), together with alphanumerics (numeric values in base 36).

- a dollar sign (`$`)
- a string of characters used for insertion operations (the "char bank")

In the example above, if we separate out the operations and convert the numbers to base 10, then we get:

```
Z :196 >1 |5=97 =31 *4 *5 +1 $x
```

Here are descriptions of the operations, where capital letters are variables:

:N
: Source text has length N (must be first op)

>N
: Final text is N (positive) characters longer than source text (must be second op)

<N
: Final text is N (positive) characters shorter than source text (must be second op)

>0
: Final text is same length as source text

+N
: Insert N characters from the bank, none of them newlines

-N
: Skip over (delete) N characters from the source text, none of them newlines

=N
: Keep N characters from the source text, none of them newlines

|L+N
: Insert N characters from the source text, containing L newlines. The last character inserted MUST be a newline, but not the (new) document's final newline.

|L-N

Delete N characters from the source text, containing L newlines. The last character inserted MUST be a newline, but not the (old) document's final newline.

|L=N

Keep N characters from the source text, containing L newlines. The last character kept MUST be a newline, and the final newline of the document is allowed.

***I**

Apply attribute I from the pool to the following `+`, `=`, `|+`, or `|=` command. In other words, any number of `*` ops can come before a `+`, `=`, or `|` but not between a `|` and the corresponding `+` or `=`. If `+`, text is inserted having this attribute. If `=`, text is kept but with the attribute applied as an attribute addition or removal. Consecutive attributes must be sorted lexically by (key,value) with key and value taken as strings. It's illegal to have duplicate keys for (key,value) pairs that apply to the same text. It's illegal to have an empty value for a key in the case of an insertion (`+`), the pair should just be omitted.

Characters from the source text that aren't accounted for are assumed to be kept with the same attributes.

Additional Constraints

- Consecutive `+`, `-`, and `=` ops of the same type that could be combined are not allowed. Whether combination is possible depends on the attributes of the ops and whether each is multiline or not. For example, two multiline deletions can never be consecutive, nor can any insertion come after a non-multiline insertion with the same attributes.
- "No-op" ops are not allowed, such as deleting 0 characters. However, attribute applications that don't have any effect are allowed.
- Characters at the end of the source text cannot be explicitly kept with no changes; if the change doesn't affect the last N characters, those "keep" ops must be left off.

- In any consecutive sequence of insertions (`+`) and deletions (`-`) with no keeps (`=`), the deletions must come before the insertions.
- The document text before and after will always end with a newline. This policy avoids a lot of special-casing of the end of the document. If a final newline is always added when importing text and removed when exporting text, then the changeset representation can be used to process text files that may or may not have a final newline.

Attribution string An *attribution string* is a series of inserts with no deletions or keeps. For example, "`*3+8|1+5`" describes the attributes of a string of length 13, where the first 8 chars have attribute 3 and the next 5 chars have no attributes, with the last of these 5 chars being a newline. Constraints apply similar to those affecting changesets, but the restriction about the final newline of the new document being added doesn't apply.

Attributes in an attribution string cannot be empty, like "`(bold,)`", they should instead be absent.

2 Further Considerations

- composing changesets/attributions with different pools.
- generalizing "applyToAttribution" to make "mutateAttributionLines" and "compose"

3 Using Unicode?

- no unicode (for efficient escaping, sightliness)
- efficient operations for ACE and collab (attributed text, etc.)
- good for time-slider
- good for API
- line-ending aware X more coherent (deleting or styling text merging with insertion)
- server-side syntax highlighting?

- unify author map with attribute pool
- unify attributed text with changeset rep
- not: reversible
- force final newline of document to be preserved

Unicode bad!

- ugly (hard to read)
- more complex to parse
- harder to store and transmit correctly
- doesn't save all that much space anyway
- blows up in size when string-escaped
- embarrassing for API