Mozilla Localization Tools

- Basic tools: Hg, SVN, Bugzilla, MXR
- Infrastructure tools: l10n dashboard, compare-locales, l10n-merge
- L10n tools: Langpacker, Silme, Koala, Verbatim, Narro, Pootle

Challenges with Mozilla l10n

- Limitations of the file types
  - <!ENTITY> XML tags
- What about plural forms?
- Declensions?
- Gender?

Mozilla L10n file types

```
# properties
offlineApps.manageUsage=Show settings

# dtd
<!ENTITY netError.search.button  "Szukaj">

# gettext
msgid "YaST installation source"
msgstr "Źródło instalacji YaST"
```

What does Mozilla do for its localization community?

Basic tools

- Hg
- SVN
- Bugzilla
- MXR

Infrastructure tools

- l10n-dashboard
- compare-locales
- l10n-merge

L10n tools

- Langpacker
- Silme
- Koala
- Verbatim
- Narro
- Pootle

What makes a good tool?

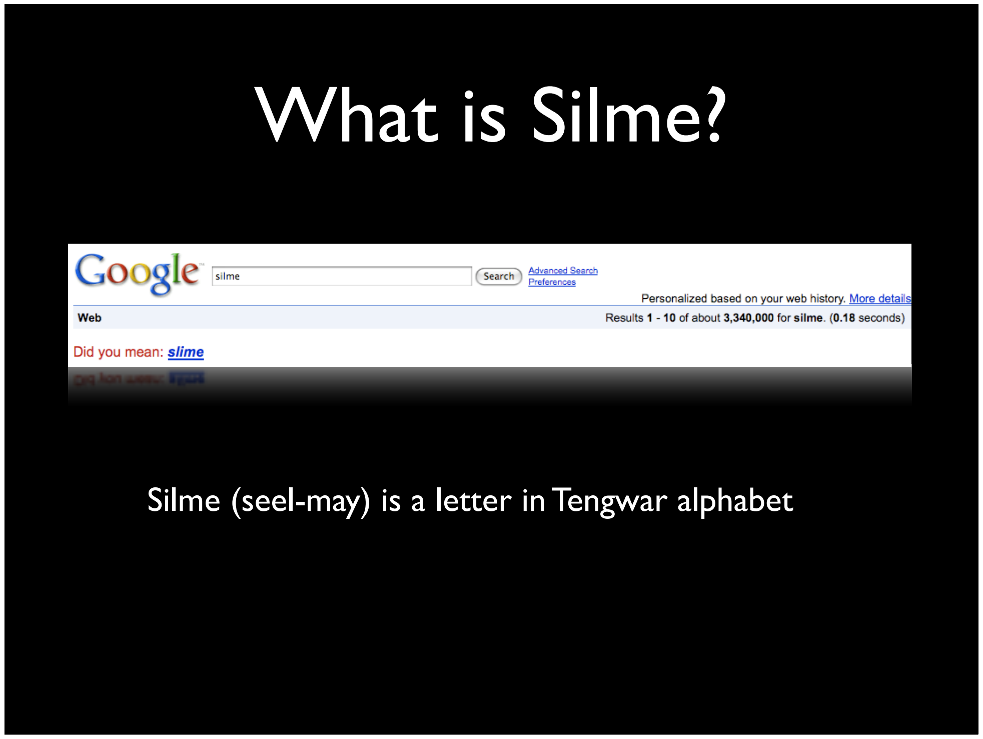
- Missing layer between apps and data
- People tend to focus on front-end l10n tools
- Tools are often strongly dependent of one data format -- DTD, .po, gettext, XLIFF, etc.
- Developers continually reinvent the wheel

Langpacker

Koala

Verbatim, Narro, Pootle

Silme



What is Silme?

silme Search Advanced Search Preferences

Personalized based on your web history. More details

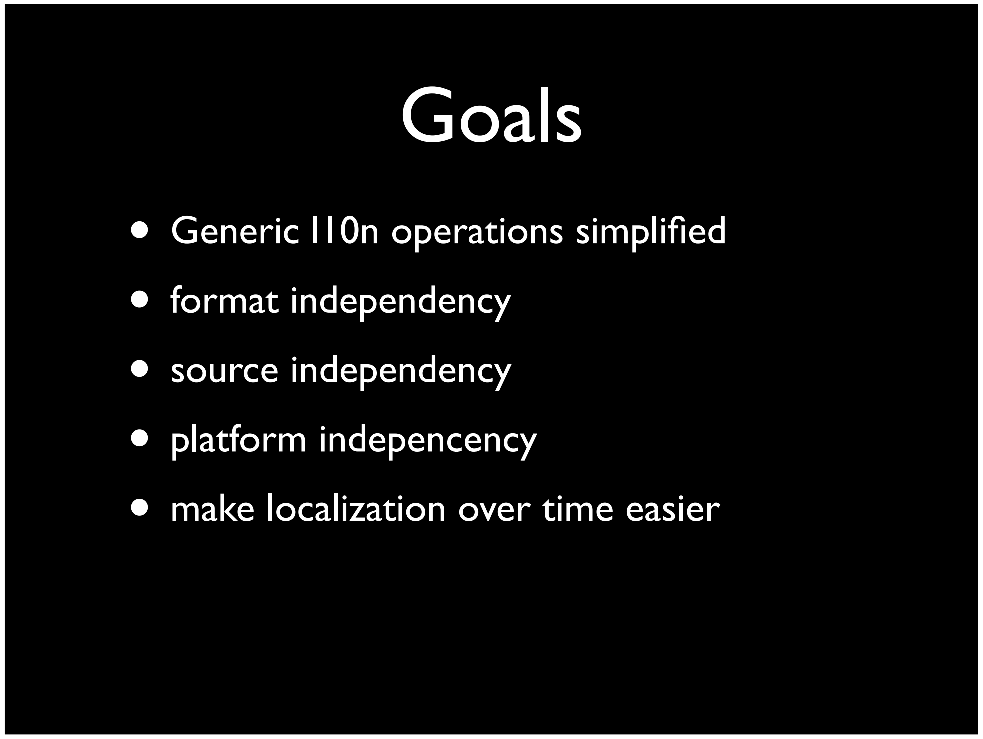
Web Results 1 - 10 of about 3,340,000 for silme. (0.18 seconds)

Did you mean: *slime*

Silme (seel-may) is a letter in Tengwar alphabet

Silme is built around several abstract concepts that allow the library to support any possible localization format, from DTD, GetText or XLIFF, to MySQL and SQLite, from JAR and normal directory to SVN, CVS or any other Revision Control System.

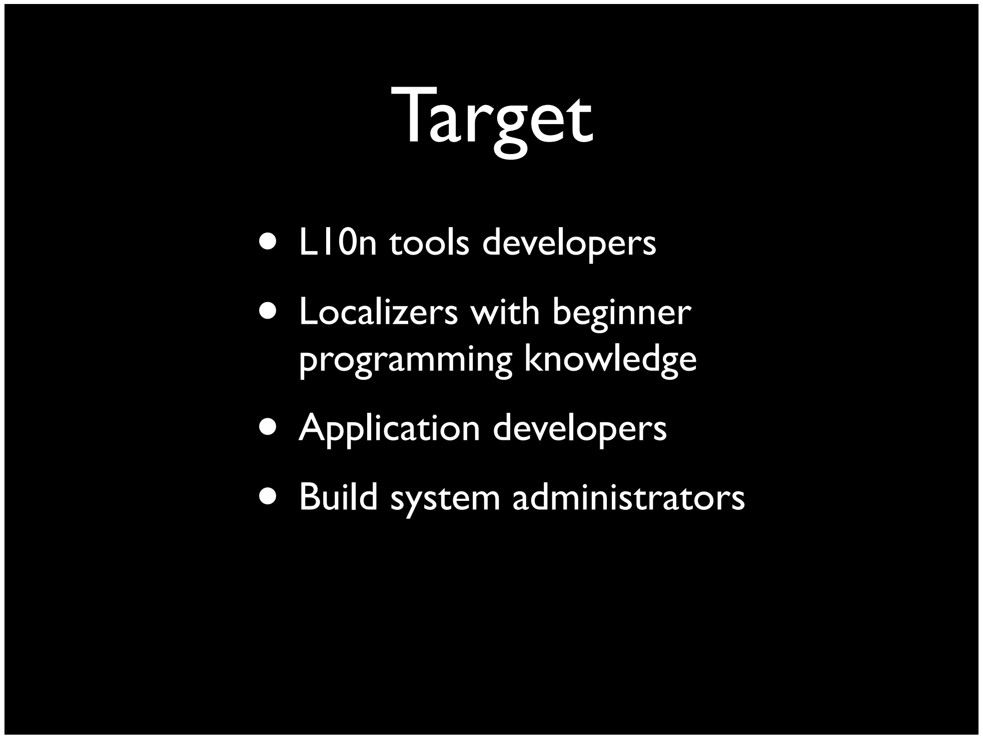
Over the next few slides, I will explain the basic concepts that will allow you to understand the architecture of the library. Of course it is just an introduction, but you'll see how we have created extensible modules for your data. Here we will focus on the simplest cases.



Goals

- Generic l10n operations simplified
- format independency
- source independency
- platform indepencency
- make localization over time easier

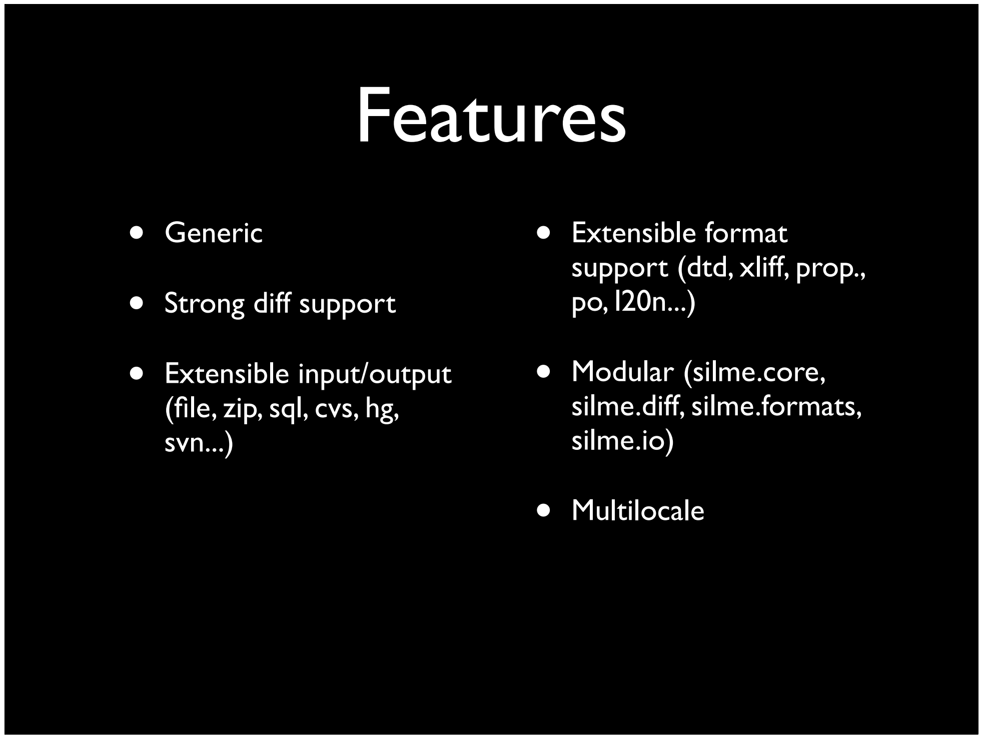
The goal of Silme is to make localization easier. That does not mean Silme is easy to understand. :) But, generally, we hope to make generic l10n operations more simple. We do this by making formats, sources and platforms independent of this tool. Essentially, the goal is to allow everyone to play. And, we hope it will make localization much easier over time.



Target

- L10n tools developers
- Localizers with beginner programming knowledge
- Application developers
- Build system administrators

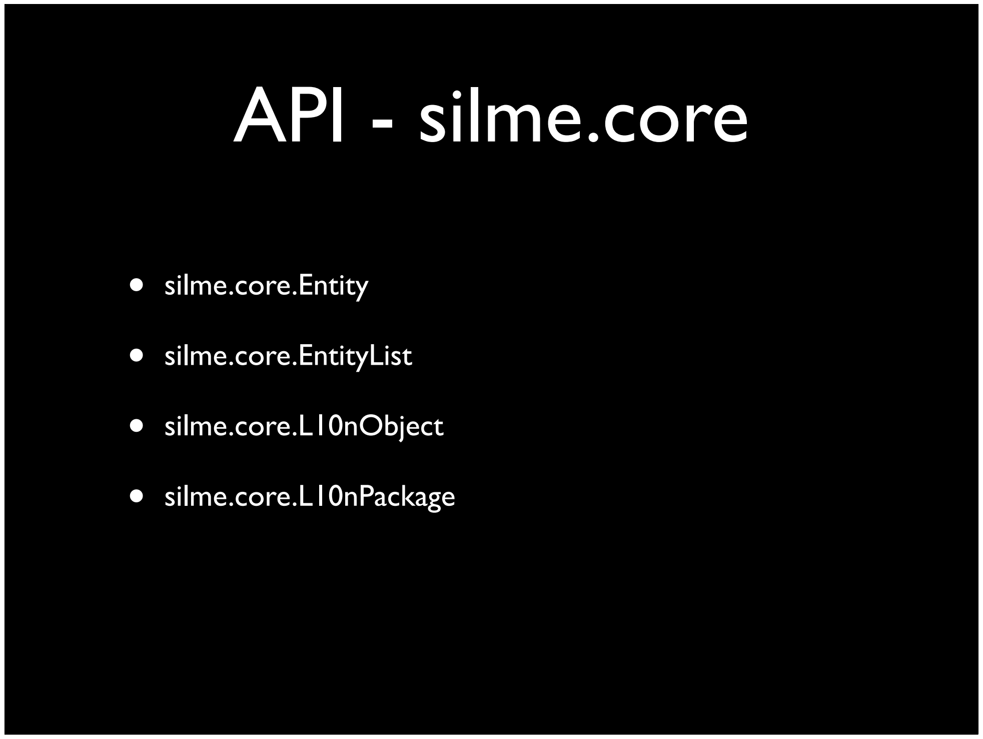
With Silme, we hope to target l10n tools developers who might use this as a middle layer between their format types and their code repository. We also want to attract localizers with beginning (or more advanced) programming knowledge to use this to construct new tools. We would love application developers to participate to build really dynamic, next generation tools. And finally, we'd like build system administrators to use in their release engineering tool inventory.



Features

- Generic
- Strong diff support
- Extensible input/output (file, zip, sql, cvs, hg, svn...)
- Extensible format support (dtd, xliff, prop., po, l20n...)
- Modular (silme.core, silme.diff, silme.formats, silme.io)
- Multilocale

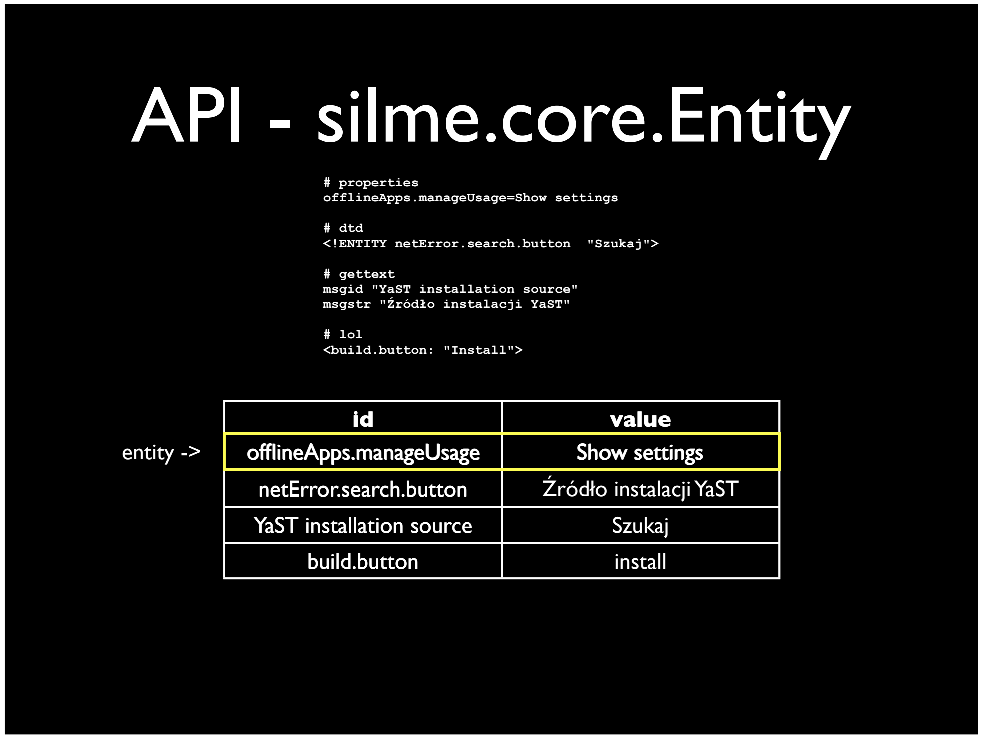
strong diff support allows you to easily show changes that have been made from one version of the localization to the next



API - silme.core

- silme.core.Entity
- silme.core.EntityList
- silme.core.L10nObject
- silme.core.L10nPackage

We have four main modules with Silme. Entity, EntityList, L10n Object, L10n Package. I’ll take you through each of these now with some examples.



API - silme.core.Entity

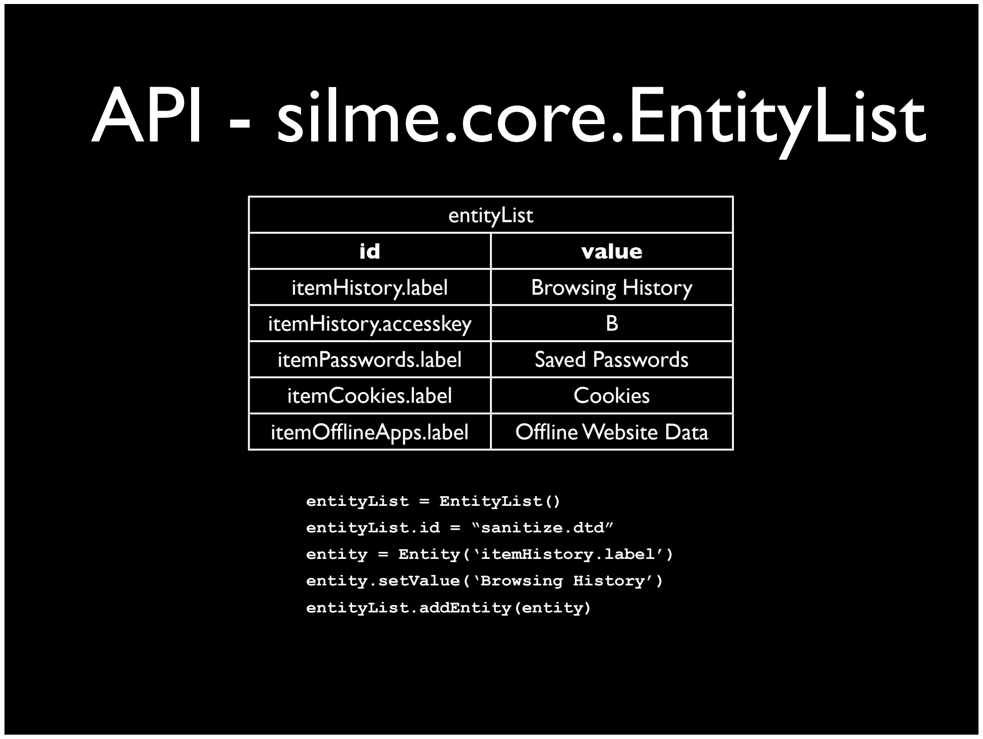
```
# properties
offlineApps.manageUsage=Show settings

# dtd
<!ENTITY netError.search.button  "Szukaj">

# gettext
msgid "YaST installation source"
msgstr "Źródło instalacji YaST"

# lol
<build.button: "Install">
```

	id	value
entity ->	offlineApps.manageUsage	Show settings
	netError.search.button	Źródło instalacji YaST
	YaST installation source	Szukaj
	build.button	install

Silme's most core and atom unit is "Entity". As some of you may know, Entity is a class that stores single pair of ID<-->VALUE in an abstract model. It is a representation of DTD's

`<!ENTITY ID "VALUE">`
Gettext's `msgid "ID"\nmsgstr "VALUE"`
MySQL's ID column and VALUE column in L10n table
etc., etc...

It's very important to understand that you can serialize any localization list to use Entity as long as you can generate a unique ID across one list and assign it a value.

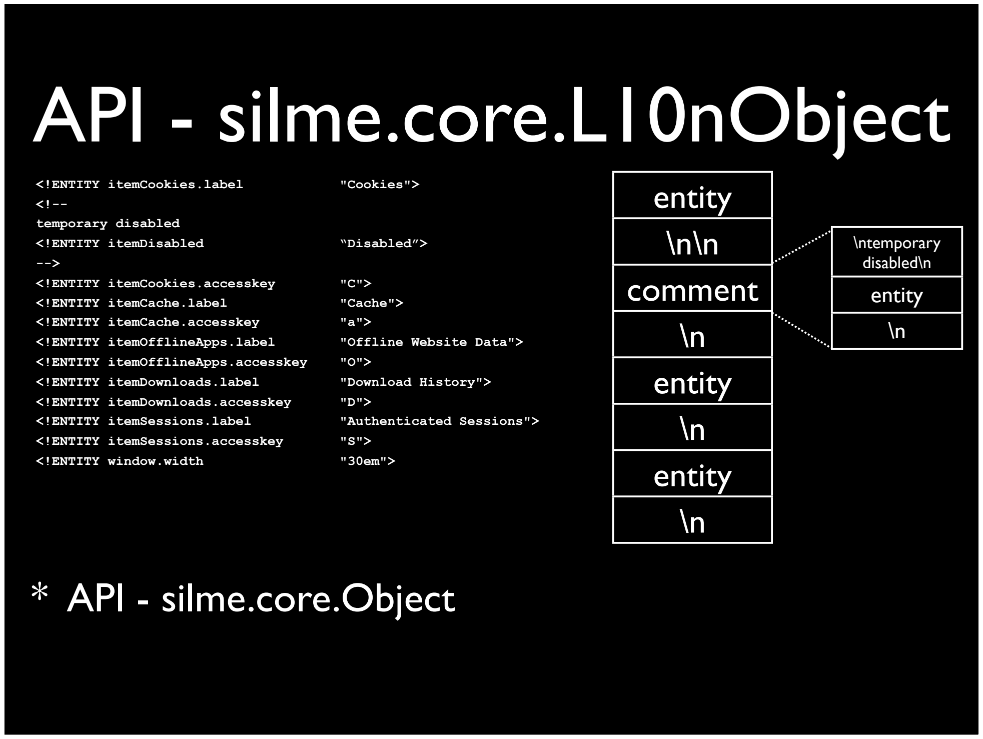


API - silme.core.EntityList

entityList	
id	**value**
itemHistory.label	Browsing History
itemHistory.accesskey	B
itemPasswords.label	Saved Passwords
itemCookies.label	Cookies
itemOfflineApps.label	Offline Website Data

```
entityList = EntityList()
entityList.id = "sanitize.dtd"
entity = Entity('itemHistory.label')
entity.setValue('Browsing History')
entityList.addEntity(entity)
```

Group of Entity objects is stored as an EntityList object. EntityList is a list (in fact, a **dict** structure in Python) that stores list of Entities and nothing more. The easiest way to imagine it is a localization SQL table containing two columns - ID and VALUE. The single row is Entity, the whole table is EntityList.



API - silme.core.L10nObject

```
<!ENTITY itemCookies.label            "Cookies">
<!--
temporary disabled
<!ENTITY itemDisabled                 “Disabled”>
-->
<!ENTITY itemCookies.accesskey        "C">
<!ENTITY itemCache.label              "Cache">
<!ENTITY itemCache.accesskey          "a">
<!ENTITY itemOfflineApps.label        "Offline Website Data">
<!ENTITY itemOfflineApps.accesskey    "O">
<!ENTITY itemDownloads.label          "Download History">
<!ENTITY itemDownloads.accesskey      "D">
<!ENTITY itemSessions.label           "Authenticated Sessions">
<!ENTITY itemSessions.accesskey       "S">
<!ENTITY window.width                 "30em">
```

* API - silme.core.Object

Above that, in some abstract sense, there is L10nObject class. L10nObject extends EntityList and is a representation of any L10n file. So besides a list of Entity objects, it also contains “comment objects” and “normal strings” between them. It's easiest to imagine it as a full representation of simple DTD file:

```
<!ENTITY myapp.title "MyApp Title">
<!--
Not used anymore
<!ENTITY title.old "Some Title">
-->
<!ENTITY notify.msg "Please, click OK to continue">
<!ENTITY notify.btn "OK">
```

will look like this:

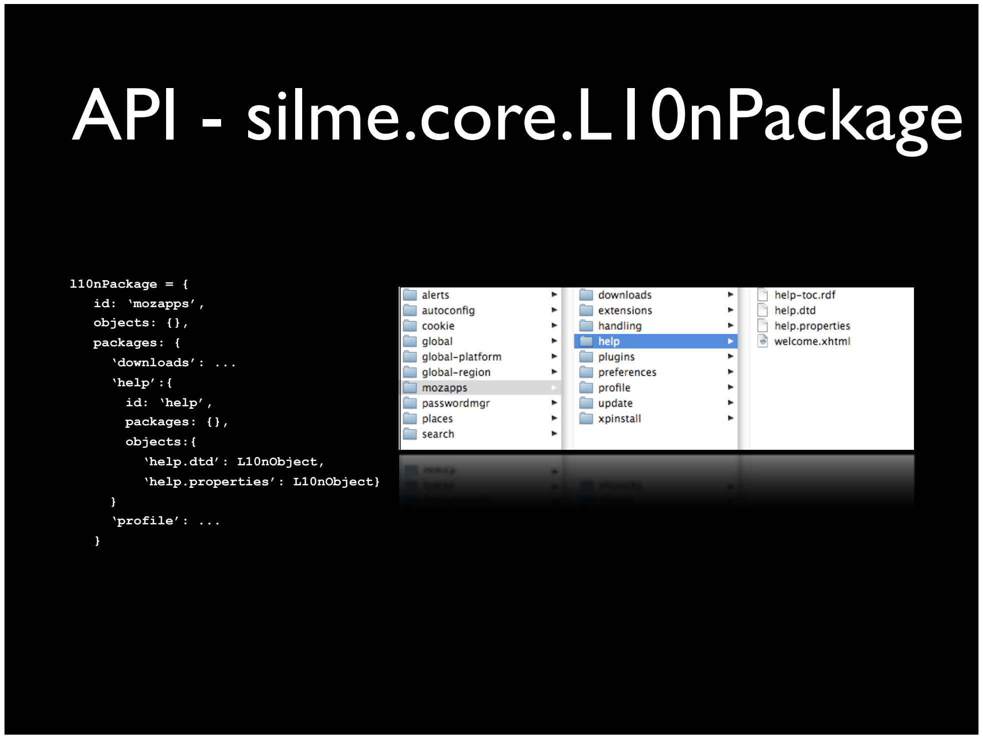
```
String('\n')
Entity(id:'myapp.title',value:'MyApp Title')
String('\n')
Comment(
  String('\nNot used anymore\n')
  Entity(id:'title.old', value:'Some Title')
)
String('\n')
Entity(id:'notify.msg',value:'Please, click OK to continue')
String('\n')
Entity(id:'notify.btn',value:'OK')
String('\n\n')
```

L10nObject is more like a file, EntityList like an SQL table. You can get EntityList out of L10nObject or you can get EntityList out of a file directly if you don't want to use the other elements of the structure. This can be handy for those who only want to use the Entities. Maybe those who are really familiar with what is going on.

The most important feature of this is that L10nObject stores whole content of the file and should always represent the full file, which means that dumping this structure back to the same format will produce identical file as a source one. In the middle you can operate, move, remove, add strings, comments and entities.

Beyond L10n Object, we have just “Object”:

Object is used to store data about files that we cannot parse. If, for example, your application will be prepared to parse DTD/PO/Properties and will get HTML file or JPEG it will store it as an Object. Object has an ID and **source** properties. Not very useful but will allow us to build a full structure above it:



API - silme.core.L10nPackage

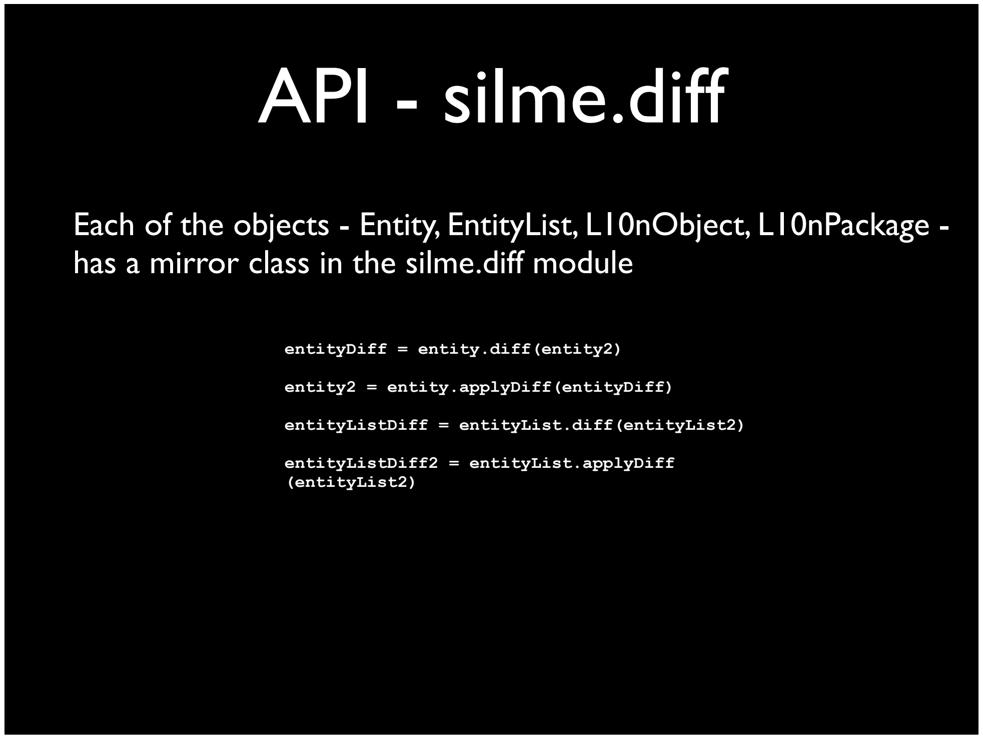
```
l10nPackage = {
   id: ‘mozapps’,
   objects: {},
   packages: {
     ‘downloads’: ...
     ‘help’:{
       id: ‘help’,
       packages: {},
       objects:{
         ‘help.dtd’: L10nObject,
         ‘help.properties’: L10nObject}
     }
     ‘profile’: ...
   }
```

alerts
autoconfig
cookie
global
global-platform
global-region
mozapps
passwordmgr
places
search
downloads
extensions
handling
help
plugins
preferences
profile
update
xpinstall
help-toc.rdf
help.dtd
help.properties
welcome.xhtml

L10nPackage is a representation of list of L10nObjects/Object/EntityLists and potentially other L10nPackages. In the file system world, the nearest similar thing is a directory. Directory can store DTD files, JPEG files, and other directories. Another similar structure is MySQL database which stores tables (EntityLists in our case).

Summary

That's all. Currently the scope of the library is to present all potential localization structures using those classes and build an API to operate on them easily. Does that seem clear. I’ll take one or two questions now.


API - silme.diff

Each of the objects - Entity, EntityList, L10nObject, L10nPackage - has a mirror class in the silme.diff module

```
entityDiff = entity.diff(entity2)

entity2 = entity.applyDiff(entityDiff)

entityListDiff = entityList.diff(entityList2)

entityListDiff2 = entityList.applyDiff
(entityList2)
```

Each and every of the objects - Entity, EntityList, L10nObject, Object, L10nPackage has it's mirror class in the Diff land.

As a result we have EntityDiff, EntityListDiff, L10nObjectDiff, ObjectDiff, L10nPackageDiff. Diff module allows you to store a difference between two objects of the same type and apply it later.

It's like a **diff** tool in Linux, but it is aware of the syntax of the files/structures and stores the diff in an appropriate way.

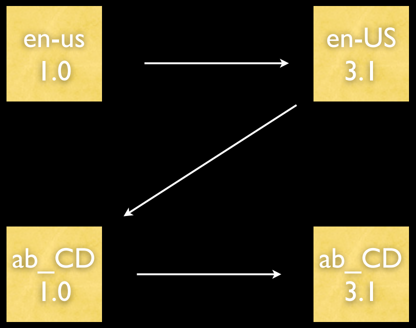
For example if a diff between two EntityLists is a value of one entity, it'll store it as EntityDiff with ID of that entity and (oldvalue,newvalue) tuple.

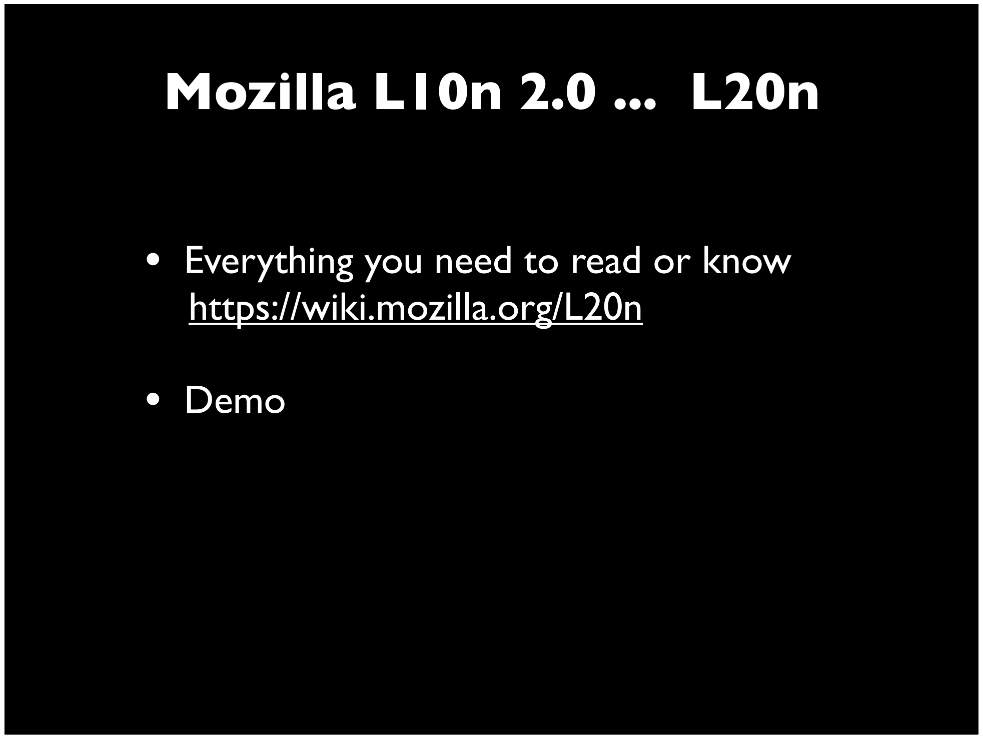
In case of an API, it'll usually go down to:

```
l10npackagediff = l10npackage1.diff(l10nPackage2)

l10npackage3.apply_diff(l10npackagediff)

l10npackage4.apply_diff(l10npackagediff)
```

but of course you will be able to manually operate on all structures by adding/removing/modifying the content of each object.

API - silme.diff: 3-way merge





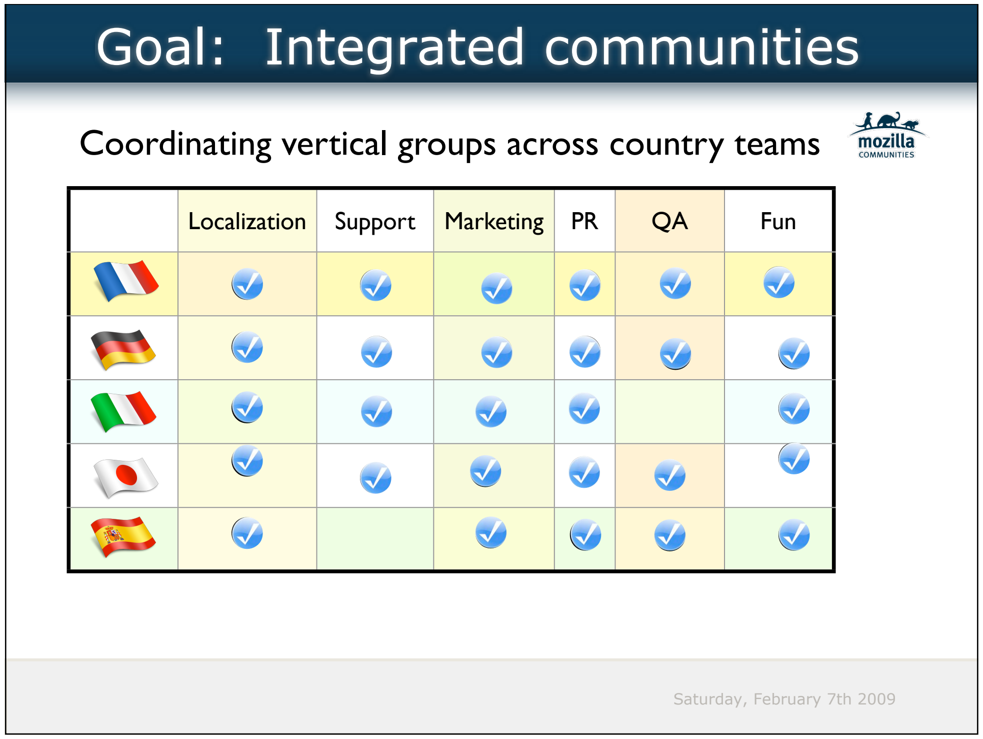
Mozilla L10n 2.0 ... L20n

- Everything you need to read or know
  https://wiki.mozilla.org/L20n
- Demo

L20n is the codename for a localization architecture taking existing approaches one step further. The name stands for l10n 2. The architecture is laid out with Mozilla applications in mind, but should be applicable to other areas as well. Implement significant changes in our l10n architecture, and this is one attempt to do that.

Mozilla Community Sites

mozilla
COMMUNITIES



Goal: Integrated communities

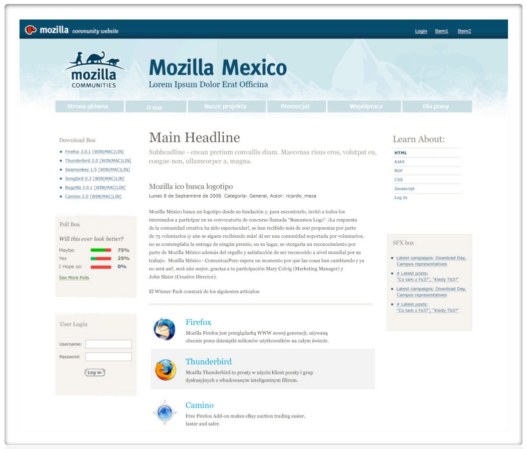
Coordinating vertical groups across country teams

	Localization	Support	Marketing	PR	QA	Fun
🇫🇷	✓	✓	✓	✓	✓	✓
🇩🇪	✓	✓	✓	✓	✓	✓
🇮🇹	✓	✓	✓	✓		✓
🇯🇵	✓	✓	✓	✓	✓	✓
🇪🇸	✓		✓	✓	✓	✓

Saturday, February 7th 2009

grid communities serve both goals by interacting on all levels, keeping local identity

mozilla communities sites



mozilla community website

Login Item1 Item2

Mozilla Mexico

Lorem Ipsum Dolor Erat Officina

Strona główna | O nas | Nasze projekty | Promocjat | Współpraca | Dla prasy

Download Box

- Firefox 3.0.1 [WIN|MAC|LIN]
- Thunderbird 2.0 [WIN|MAC|LIN]
- Seamonkey 1.5 [WIN|MAC|LIN]
- Songbird 0.3 [WIN|MAC|LIN]
- Bugzilla 3.0.1 [WIN|MAC|LIN]
- Camino 2.0 [WIN|MAC|LIN]

Poll Box

Will this ever look better?

Maybe:	75%
Yes	25%
I Hope so:	0%

See More Polls

User Login

Username:

Password:

Log in

Main Headline

Subheadline - enean pretium convallis diam. Maecenas risus eros, volutpat eu, congue non, ullamcorper a, magna.

Mozilla ico busca logotipo

Lunes 8 de Septiembre de 2008. Categoría: General, Autor: ricardo_meza

Mozilla México busca un logotipo desde su fundación y, para encontrarlo, invitó a todos los interesados a participar en su convocatoria de concurso llamada "Buscamos Logo". ¡La respuesta de la comunidad creativa ha sido espectacular!, se han recibido más de 100 propuestas por parte de 75 voluntarios ¡y aún se siguen recibiendo más! Al ser una comunidad soportada por voluntarios, no se contemplaba la entrega de ningún premio, en su lugar, se otorgaría un reconocimiento por parte de Mozilla México además del orgullo y satisfacción de ser reconocido a nivel mundial por su trabajo. Mozilla México - Comunica¡Pero espera un momento por que las cosas han cambiando y ya no será así!, será aún mejor, gracias a tu participación Mary Colvig (Marketing Manager) y John Slater (Creative Director).

El Winner Pack constará de los siguientes artículos:

Firefox

Mozilla Firefox jest przeglądarką WWW nowej generacji, używaną obecnie przez dziesiątki milionów użytkowników na całym świecie.

Thunderbird

Mozilla Thunderbird to prosty w użyciu klient poczty i grup dyskusyjnych z wbudowanym inteligentnym filtrem.

Camino

Free Firefox Add-on makes eBay auction trading easier, faster and safer.

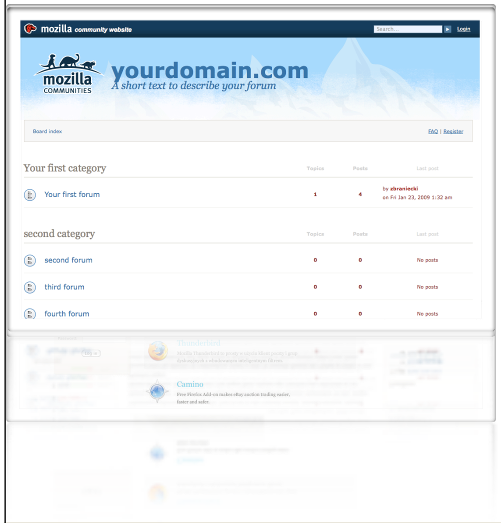
Learn About:

- HTML
- AJAX
- RDF
- CSS
- Javascript
- Log In

SFX box

- Latest campaigns: Download Day, Campus representatives
- # Latest posts: "Co tam z Fx3?", "Kiedy Tb3?"
- Latest campaigns: Download Day, Campus representatives
- # Latest posts: "Co tam z Fx3?", "Kiedy Tb3?"

mozilla communities sites


mozilla community website

Search... Login

yourdomain.com

A short text to describe your forum

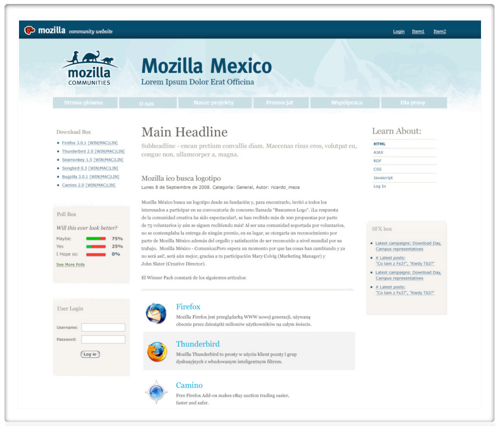
Board index FAQ | Register

Your first category	Topics	Posts	Last post
Your first forum	1	4	by zbraniecki on Fri Jan 23, 2009 1:32 am

second category	Topics	Posts	Last post
second forum	0	0	No posts
third forum	0	0	No posts
fourth forum	0	0	No posts

- Main page
- News
- Wiki - Mediawiki
- Forum - punBB, phpBB
- Blogs - Wordpress
- Planet

mozilla communities widgets



Mozilla Mexico
Lorem Ipsum Dolor Erat Officina

Download Box
- Firefox 3.0.1 [WIN|MAC|LIN]
- Thunderbird 2.0 [WIN|MAC|LIN]
- Seamonkey 1.5 [WIN|MAC|LIN]
- Songbird 0.3 [WIN|MAC|LIN]
- Bugzilla 3.0.1 [WIN|MAC|LIN]
- Camino 2.0 [WIN|MAC|LIN]

Poll Box

Will this ever look better?

Maybe:	75%
Yes	25%
I Hope so:	0%

See More Polls

User Login

Username:

Password:

Main Headline

Subheadline - enean pretium convallis diam. Maecenas risus eros, volutpat eu, congue non, ullamcorper a, magna.

Mozilla ico busca logotipo

Lunes 8 de Septiembre de 2008. Categoría: General, Autor: ricardo_meza

Mozilla México busca un logotipo desde su fundación y, para encontrarlo, invitó a todos los interesados a participar en su convocatoria de concurso llamada "Buscamos Logo". ¡La respuesta de la comunidad creativa ha sido espectacular!, se han recibido más de 100 propuestas por parte de 75 voluntarios ¡y aún se siguen recibiendo más! Al ser una comunidad soportada por voluntarios, no se contemplaba la entrega de ningún premio, en su lugar, se otorgaría un reconocimiento por parte de Mozilla México además del orgullo y satisfacción de ser reconocido a nivel mundial por su trabajo. Mozilla México - Comunica¡Pero espera un momento por que las cosas han cambiando y ya no será así!, será aún mejor, gracias a tu participación Mary Colvig (Marketing Manager) y John Slater (Creative Director).

El Winner Pack constará de los siguientes artículos:

Firefox
Mozilla Firefox jest przeglądarką WWW nowej generacji, używaną obecnie przez dziesiątki milionów użytkowników na całym świecie.

Thunderbird
Mozilla Thunderbird to prosty w użyciu klient poczty i grup dyskusyjnych z wbudowanym inteligentnym filtrem.

Camino
Free Firefox Add-on makes eBay auction trading easier, faster and safer.

Learn About:
- HTML
- AJAX
- RDF
- CSS
- Javascript
- Log In

SFX box
- Latest campaigns: Download Day, Campus representatives
- # Latest posts: "Co tam z Fx3?", "Kiedy Tb3?"
- Latest campaigns: Download Day, Campus representatives
- # Latest posts: "Co tam z Fx3?", "Kiedy Tb3?"

Mozilla Community Logo

- Layer between communities and our official branding
- Unifying element
- Liberal licensing - do what you want

mozilla
COMMUNITIES

Mozilla Community Theme

- Very clean code base, HTML 5 ready

```
<html>
  <div class="header">
  <div class="middle">
    <div class="aside" id="left-bar" />
    <div class="section" id="content" />
    <div class="aside" id="right-bar" />
  </div>
  <div class="footer">
</html>
```

$$\begin{tikzcd}
\text{core.css} & \\
\text{skeleton.css} & \text{header-bar.css} \\
\text{layout.css} \arrow[ur] \arrow[r] \arrow[dr] \arrow[ddr] & \text{header.css} \\
\text{rtl.css} & \text{content.css} \\
 & \text{footer.css} \\
\text{custom.css} \arrow[r] & \text{color.css}
\end{tikzcd}$$

- OpenID, Single sign on
- Best practices from communities

Benefits of the MCS

- Small community website
- Out-of-the-box experience
- Easy maintenance
- Easy to extend

MCS uses Drupal

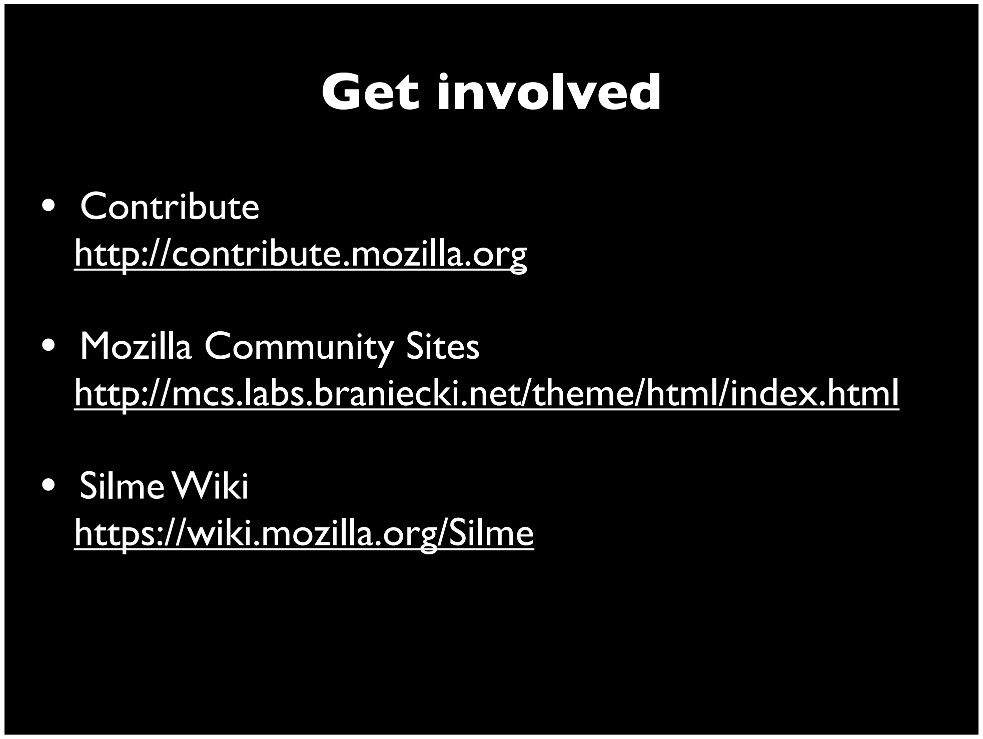
- powerful community system
- blogs, news, forums, calendar, wiki, etc.
- able to handle major communities, tons of extensions

MCS future

- Launch MCS in communities, including India
  - MozCampDelhi
- Get all applications ready
- Common web admin panel
- Keep working on implementation (JS, CSS, HTML etc.)
- Cherry-pick best extensions and include them

future

- T-shirts
- MCS Slide Templates
- Style Guide, extend the style
- Community aggregators
- Spin-offs?



Get involved

- Contribute
  http://contribute.mozilla.org
- Mozilla Community Sites
  http://mcs.labs.braniecki.net/theme/html/index.html
- Silme Wiki
  https://wiki.mozilla.org/Silme

That's all for now. This article explained the basic concepts behind the library and I hope you'll find the library useful enough to experiment with writing apps on top of it and/or working with the library itself.

Questions?

sethb@mozilla.com
http://blog.mozilla.com/seth