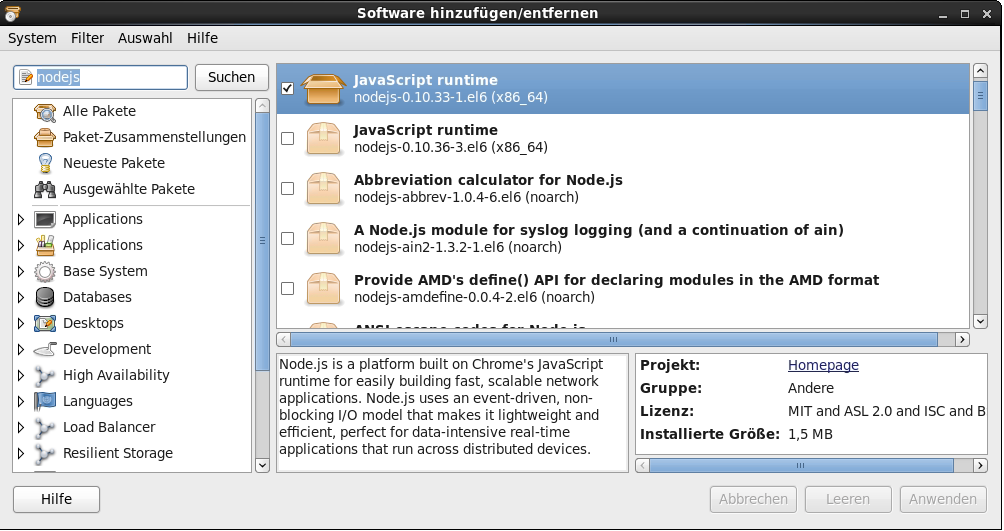
NodeJS



Parsoid Service

from GitHub <https://github.com/wikimedia/parsoid/blob/master/package.json>

* version 0.2.0-git
* checked out to **/usr/share/parsoid** on my CentOS6
* navigate to checked out folder and run **npm install**
* tweaking the **api/localsettings.js**

…

parsoidConfig.setInterwiki( 'localhost', 'http://localhost/w/api.php' );

…

1 param: changed to to ‘**CBASEalmUUID’** from VisualEditor settings (see later)

2 param: pointing to **one** of my mediawiki instances from Tuleap **…wiki/…/api.php**

**CHANLLENGE: The Parsoid Service should be able to support all Tuleap-Mediawiki instances?! It’s possible to set multiple SetInterwikis…**

* test the parsoid service through **api/runserver.sh**

Parsoid Mediawiki Extension

<http://www.mediawiki.org/wiki/Extension:Parsoid>

* Install to **/usr/share/mediawiki-tuleap-123/extensions**
* **LocalSettings.php**

*// Parsoid PHP Extension*

*require\_once "$IP/extensions/Parsoid/Parsoid.php";*

Visual Editor

<http://www.mediawiki.org/wiki/Special:ExtensionDistributor?extdist_name=VisualEditor&extdist_version=REL1_23&extdist_submit=Continue>

**LocalSettings.php:**

// Visual Editor Extension

require\_once "$IP/extensions/VisualEditor/VisualEditor.php";

// Enable by default for everybody

$wgDefaultUserOptions['visualeditor-enable'] = 1;

// Don't allow users to disable it

$wgHiddenPrefs[] = 'visualeditor-enable';

// OPTIONAL: Enable VisualEditor's experimental code features

$wgDefaultUserOptions['visualeditor-enable-experimental'] = 1;

// URL to the Parsoid instance

// MUST NOT end in a slash due to Parsoid bug

**$wgVisualEditorParsoidURL =** **'http://localhost:8000';**

// Proxy to use for curl requests.

// false: use direct connection to Parsoid daemon ($wgHTTPProxy is not used

// either)

$wgVisualEditorParsoidHTTPProxy = false;

// Interwiki prefix to pass to the Parsoid instance

// Parsoid will be called as $url/$prefix/$pagename

**$wgVisualEditorParsoidPrefix =** **'CBASEalmUUID'; //See Parsoid Settings**

**//CHALLENGE: We have only one LocalSettings.php for all Mediawiki instances ?! Tuleap should handle that somehow (e.g. Project Short Name as Prefix )**

// Forward users' Cookie: headers to Parsoid. Required for private wikis (login required to read).

// If the wiki is not private (i.e. $wgGroupPermissions['\*']['read'] is true) this configuration

// variable will be ignored.

//

// NOTE: This feature requires a non-locking session store. The default session store will not work

// and will cause deadlocks when trying to use this feature, causing you to receive "MWHttpRequest"

// errors. If you have a private wiki and are using the default session store in Apache, be sure to

// uncomment this line.

$wgSessionsInObjectCache = true;

//

// WARNING: ONLY enable forwarding users' cookies on private wikis and ONLY IF you understand the

// SECURITY IMPLICATIONS of sending Cookie headers to Parsoid over HTTP. For security reasons, it is

// strongly recommended that $wgVisualEditorParsoidURL be pointed to localhost if this setting is

// enabled.

**$wgVisualEditorParsoidForwardCookies = true;//CHALLENGE: Private wikis in Tuleap was not possible cause the parsoid service is redirected to Tuleaps Login Page when requesting the mediawiki instance api.php. Even the ForwardCookies not helped. The only way it worked was changing project to public.**

// Timeout for HTTP requests to Parsoid in seconds

$wgVisualEditorParsoidTimeout = 30;

// Serialization cache timeout, in seconds

$wgVisualEditorSerializationCacheTimeout = 3600;

// Namespaces to enable VisualEditor in

$wgVisualEditorNamespaces = array\_merge( $wgContentNamespaces, array( NS\_USER ) );

// Whether to enable the (experimental for now) TOC widget

$wgVisualEditorEnableTocWidget = false;

// List of skins VisualEditor integration supports

**$wgVisualEditorSupportedSkins = array( 'vector', 'apex', 'monobook', 'minerva', 'tuleap123');**

// Whether to use change tagging for VisualEditor edits

$wgVisualEditorUseChangeTagging = true;

// Whether to disable for logged-in users

// This allows you to enable the 'visualeditor-enable' preference by default

// but still disable VE for logged-out users (by setting this to false).

$wgVisualEditorDisableForAnons = false;

// Whether to show the "welcome to the beta" dialog the first time a user uses VisualEditor

$wgVisualEditorShowBetaWelcome = false;

// Where to put the VisualEditor edit tab

// 'before': put it right before the old edit tab

// 'after': put it right after the old edit tab

$wgVisualEditorTabPosition = 'before';

$wgVisualEditorTabMessages = array(

// i18n message key to use for the VisualEditor edit tab

// If null, the default edit tab caption will be used

// The 'visualeditor-ca-ve-edit' message is available for this

'edit' => null,

// i18n message key to use for the old edit tab

// If null, the tab's caption will not be changed

'editsource' => 'visualeditor-ca-editsource',

// i18n message key to use for the VisualEditor create tab

// If null, the default create tab caption will be used

// The 'visualeditor-ca-ve-create' message is available for this

'create' => null,

// i18n message key to use for the old create tab

// If null, the tab's caption will not be changed

'createsource' => 'visualeditor-ca-createsource',

// i18n message key to use for the VisualEditor section edit link

// If null, the default edit section link caption will be used

'editsection' => null,

// i18n message key to use for the source section edit link

// If null, the link's caption will not be changed

'editsectionsource' => 'visualeditor-ca-editsource-section',

// i18n message key for an optional appendix to add to each of these from JS

// Use this if you have HTML messages to add

// The 'visualeditor-beta-appendix' message is available for this purpose

'editappendix' => null,

'editsourceappendix' => null,

'createappendix' => null,

'createsourceappendix' => null,

'editsectionappendix' => null,

'editsectionsourceappendix' => null,

);