Trac Links

Table of Contents

TracLinks are a fundamental feature of Trac, because they allow easy hyperlinking between the various entities in the system — such as tickets, reports, changesets, Wiki pages, milestones, and source files — from anywhere where WikiFormatting is used.

TracLinks are generally of the form **type:id** (where *id* represents the number, name or path of the item) though some frequently used kinds of items also have short-hand notations.

Where to use TracLinks

You can use TracLinks in:

- Source code (Subversion) commit messages
- Wiki pages
- Full descriptions for tickets, reports and milestones

and any other text fields explicitly marked as supporting WikiFormatting.

Overview

Wiki Markup	Display

Wiki pages
`CamelCase` or `wiki:CamelCase`
Parent page
`[..]`
Tickets
`#1` or `ticket:1`
Ticket comments
`comment:1:ticket:2`
Reports
`{1}` or `report:1`
Milestones
`milestone:1.0`
Attachment
`attachment:example.tgz` (for current page attachment), `attachment:attachment.1073.diff:ticket:944` (absolute path)
Changesets
`r1`, `[1]`, `changeset:1` or (restricted) `[1/trunk]`, `changeset:1/trunk`, `[1/repository]`
Revision log
`r1:3`, `[1:3]` or `log:@1:3`, `log:trunk@1:3`, `[2:5/trunk]`
Diffs
`diff:@1:3`, `diff:plugins/0.12/mercurial-plugin@9128:9953`, `diff:tags/trac-0.9.2/wiki-default//tags/trac-0.9.3/wiki-default` or `diff:trunk/trac@3538//sandbox/vc-refactoring@3539`
Files
`source:trunk/COPYING`, `source:/trunk/COPYING@200` (at version 200), `source:/trunk/COPYING@200#L25` (at version 200, line 25)
	Wiki pages
	CamelCase or wiki:CamelCase
	Parent page
	..
	Tickets
	#1 or ticket:1
	Ticket comments
	comment:1:ticket:2
	Reports
	{1} or report:1
	Milestones
	milestone:1.0
	Attachment
	attachment:example.tgz (for current page attachment), attachment:attachment.1073.diff:ticket:944 (absolute path)
	Changesets
	r1, [1], changeset:1 or (restricted) [1/trunk], changeset:1/trunk, [1/repository]
	Revision log
	r1:3, [1:3] or log:@1:3, log:trunk@1:3, [2:5/trunk]
	Diffs
	diff:@1:3, diff:plugins/0.12/mercurial-plugin@9128:9953, diff:tags/trac-0.9.2/wiki-default//tags/trac-0.9.3/wiki-default or diff:trunk/trac@3538//sandbox/vc-refactoring@3539
	Files
	source:trunk/COPYING, source:/trunk/COPYING@200 (at version 200), source:/trunk/COPYING@200#L25 (at version 200, line 25)

Note: The wiki:CamelCase form is rarely used, but it can be convenient to refer to pages whose names do not follow WikiPageNames rules, ie single words, non-alphabetic characters, etc. See WikiPageNames for more about features specific to links to Wiki page names.

TracLinks are a very simple idea, but actually allow quite a complex network of information. In practice, it's very intuitive and simple to use, and we've found the "link trail" extremely helpful to better understand what's happening in a project or why a particular change was made.

Advanced use of TracLinks

Relative links

To create a link to a ■SubWiki-page just use a '/':

```
 WikiPage/SubWikiPage or ./SubWikiPage
```

To link from a ■SubWiki page to a parent, simply use a '..':

```
  [..] or [[..]]
```

.. or ..

To link from a ■SubWiki page to a sibling page, use a '../':

```
  [../Sibling see next sibling] or [[../Sibling|see next sibling]]
```

see next sibling? or see next sibling?

But in practice you often won't need to add the `../` prefix to link to a sibling page. For resolving the location of a wiki link, it's the target page closest in the hierarchy to the page where the link is written which will be selected. So for example, within a sub-hierarchy, a sibling page will be targeted in preference to a toplevel page. This makes it easy to copy or move pages to a sub-hierarchy by renaming without having to adapt the links.

To link explicitly to a toplevel Wiki page, use the `wiki:/` prefix. Be careful **not** to use the `/` prefix alone, as this corresponds to the #Server-relativelinks syntax and with such a link you will lack the `/wiki/` part in the resulting URL. A link such as `[../newticket]` will stay in the wiki namespace and therefore link to a sibling page.

Link anchors

To create a link to a specific anchor in a page, use '#':

```
  [#Linkanchors Link anchors] or [[#Linkanchors|Link anchors]]
```

Link anchors or Link anchors

Hint: when you move your mouse over the title of a section, a '¶' character will be displayed. This is a link to that specific section and you can use this to copy the `#...` part inside a relative link to an anchor.

To create a link to the first or last occurrence of a term on a page, use a *pseudo anchor* starting with '#/' or '#?':

```
  [#/Milestone first occurrence of Milestone] or
 [#?Milestone last occurrence of Milestone]
```

first occurrence of Milestone or last occurrence of Milestone

This will also highlight all other matches on the linked page. By default only case sensitive matches are considered. To include case insensitive matches append '/i':

```
  [#/Milestone/i first occurrence of Milestone or milestone] or
 [#?Milestone/i last occurrence of Milestone or milestone]
```

first occurrence of Milestone or milestone or last occurrence of Milestone or milestone

(since Trac 1.0)

Such anchors can be very useful for linking to specific lines in a file in the source browser:

```
 [trac:source:tags/trac-0.12/trac/wiki/api.py#L127 Line 127] or
[trac:source:tags/trac-0.12/trac/ticket/roadmap.py#L47 Line 47]
```

■Line 127 or ■Line 47

(Hint: The line numbers displayed in the source browser are links to anchors on the respective lines.)

Since such links become outdated when the file changes, it can be useful to link using a '#/' pseudo anchor instead:

```
 [trac:source:trunk/trac/wiki/api.py#/IWikiSyntaxProvider IWikiSyntaxProvider] or
[trac:source:trunk/trac/env.py#/ISystemInfoProvider ISystemInfoProvider]
```

■IWikiSyntaxProvider or ■ISystemInfoProvider

InterWiki links

Other prefixes can be defined freely and made to point to resources in other Web applications. The definition of those prefixes as well as the URLs of the corresponding Web applications is defined in a special Wiki page, the InterMapTxt page. Note that while this could be used to create links to other Trac environments, there is a more specialized way to register other Trac environments which offers greater flexibility.

InterTrac links

This can be seen as a kind of InterWiki link specialized for targeting other Trac projects.

Any type of Trac link can be written in one Trac environment and actually refer to resources in another Trac environment. All that is required is to prefix the Trac link with the name of the other Trac environment followed by a colon. The other Trac environment must be registered on the InterTrac page.

A distinctive advantage of InterTrac links over InterWiki links is that the shorthand form of Trac links (e.g. `{}`, `r`, #) can also be used. For example if T was set as an alias for Trac, links to Trac tickets can be written #T234, links to Trac changesets can be written ■[trac 1508]. See InterTrac for the complete details.

Server-relative links

It is often useful to be able to link to objects in your project that have no built-in Trac linking mechanism, such as static resources, `newticket`, a shared `/register` page on the server, etc.

To link to resources inside the project, use either an absolute path from the project root, or a relative link from the URL of the current page (*Changed in 0.11*):

```
[/newticket Create a new ticket] or [[//newticket|Create a new ticket]]
[/ home] or [[/|home]]
```

Display: Create a new ticket or Create a new ticket home or home

To link to another location on the server (possibly outside the project but on the same host), use the `//` prefix (*Changed in 0.11*):

```
[//register Register Here] or [[//register|Register Here]]
```

Display: Register Here or Register Here

Quoting space in TracLinks

Immediately after a TracLinks prefix, targets containing space characters should be enclosed in a pair of quotes or double quotes. Examples:

- wiki:"The whitespace convention"
- attachment:'the file.txt' or
- attachment:"the file.txt"
- attachment:"the file.txt:ticket:123"

Note that by using ■WikiCreole style links, it's quite natural to write links containing spaces:

- [[The whitespace convention]]

- [[attachment:the file.txt]]

Escaping Links

To prevent parsing of a TracLink, you can escape it by preceding it with a '!' (exclamation mark).

```
 !NoLinkHere.
![42] is not a link either.
```

Display:

> NoLinkHere. [42] is not a link either.

Parameterized Trac links

Many Trac resources have more than one way to be rendered, depending on some extra parameters. For example, a Wiki page can accept a `version` or a `format` parameter, a report can make use of dynamic variables, etc.

Trac links can support an arbitrary set of parameters, written in the same way as they would be for the corresponding URL. Some examples:

- `wiki:WikiStart?format=txt`
- `ticket:1?version=1`
- `[/newticket?component=module1 create a ticket for module1]`
- `[/newticket?summary=Add+short+description+here create a ticket with URL with spaces]`

TracLinks Reference

The following sections describe the individual link types in detail, as well as notes on advanced usage of links.

attachment: links

The link syntax for attachments is as follows:

- attachment:the_file.txt creates a link to the attachment the_file.txt of the current object
- attachment:the_file.txt:wiki:MyPage creates a link to the attachment the_file.txt of the MyPage wiki page
- attachment:the_file.txt:ticket:753 creates a link to the attachment the_file.txt of the ticket 753

Note that the older way, putting the filename at the end, is still supported: attachment:ticket:753:the_file.txt, but is not recommended.

If you'd like to create a direct link to the content of the attached file instead of a link to the attachment page, simply use `raw-attachment:` instead of `attachment:`.

This can be useful for pointing directly to an HTML document, for example. Note that for this use case, you'd have to allow the web browser to render the content by setting `[attachment] render_unsafe_content = yes` (see TracIni#attachment-section). Caveat: only do that in environments for which you're 100% confident you can trust the people who are able to attach files, as otherwise this would open up your site to ■cross-site scripting attacks.

See also #export:links.

comment: links

When you're inside a given ticket, you can simply write e.g. comment:3 to link to the third change comment. It is possible to link to a comment of a specific ticket from anywhere using one of the following syntax:

- `comment:3:ticket:123`
- `ticket:123#comment:3` (note that you can't write `#123#!comment:3`!)

It is also possible to link to the ticket's description using one of the following syntax:

- `comment:description` (within the ticket)
- `comment:description:ticket:123`
- `ticket:123#comment:description`

htdocs: links

Use `htdocs:path/to/file` to reference files in the `htdocs` directory of the Trac environment, the web resource directory.

query: links

See TracQuery#UsingTracLinks and #ticket:links.

search: links

See TracSearch#SearchLinks

ticket: links

aliases: `bug:`, `issue:`

Besides the obvious `ticket:id` form, it is also possible to specify a list of tickets or even a range of tickets instead of the `id`. This generates a link to a custom query view containing this fixed set of tickets.

Example:

- `ticket:5000-6000`
- `ticket:1,150`

timeline: links

Links to the timeline can be created by specifying a date in the ■ISO:8601 format. The date can be optionally followed by a time specification. The time is interpreted as being UTC time, but if you don't want to compute the UTC time, you can specify a local time followed by your timezone offset relative to UTC.

Examples:

- `timeline:2008-01-29`
- `timeline:2008-01-29T15:48`
- `timeline:2008-01-29T15:48Z`
- `timeline:2008-01-29T16:48+01`
- `timeline:2008-01-29T16:48+0100`
- `timeline:2008-01-29T16:48+01:00`

wiki: links

See WikiPageNames and quoting space in TracLinks above. It is possible to create a link to a specific page revision using the syntax WikiStart@1.

Version Control related links

It should be noted that multiple repository support works by creating a kind of virtual namespace for versioned files in which the toplevel folders correspond to the repository names. Therefore, in presence of multiple repositories, a */path* specification in the syntax of links detailed below should start with the name of the repository. If omitted, the default repository is used. In case a toplevel folder of the default repository has the same name as a repository, the latter "wins". One can always access such folder by fully qualifying it. The default repository can be an alias of a named repository, or conversely, it is always possible to create an alias for the default repository, ask your Trac administrator.

For example, `source:/trunk/COPYING` targets the path `/trunk/COPYING` in the default repository, whereas `source:/projectA/trunk/COPYING` targets the path `/trunk/COPYING` in the repository named `projectA`. This can be the same file if `'projectA'` is an alias to the default repository or if `''` (the default repository) is an alias to `'projectA'`.

source: links

aliases: `browser:`, `repos:`

The default behavior for a `source:/some/path link` is to open the browser in that directory directory if the path points to a directory or to show the latest content of the file.

It's also possible to link directly to a specific revision of a file like this:

- `source:/some/file@123` - link to the file's revision 123
- `source:/some/file@head` - link explicitly to the latest revision of the file
- `source:/some/file@named-branch` - link to latest revision of the specified file in `named-branch` (DVCS such as Git or Mercurial)

If the revision is specified, one can even link to a specific line number:

- `source:/some/file@123#L10`
- `source:/tag/0.10@head#L10`
- `source:/some/file@named-branch#L10`

Finally, one can also highlight an arbitrary set of lines:

- `source:/some/file@123:10-20,100,103#L99` - highlight lines 10 to 20, and lines 100 and 103, and target line 99
- or without version number (the `@` is still needed): `source:/some/file@:10-20,100,103#L99`. Version can be omitted when the path is pointing to a source file that will no longer change (like `source:/tags/...`), otherwise it's better to specify which lines of *which version* of the file you're talking about.

Note that in presence of multiple repositories, the name of the repository is simply integrated in the path you specify for `source:` (e.g. `source:reponame/trunk/README`). *(since 0.12)*

export: links

To force the download of a file in the repository, as opposed to displaying it in the browser, use the `export` link. Several forms are available:

- `export:/some/file` - get the HEAD revision of the specified file
- `export:123:/some/file` - get revision 123 of the specified file
- `export:/some/file@123` - get revision 123 of the specified file
- `export:/some/file@named-branch` - get latest revision of the specified file in `named-branch` (DVCS such as Git or Mercurial).

This can be very useful for displaying XML or HTML documentation with correct stylesheets and images, in case that has been checked in into the repository. Note that for this use case, you'd have to allow the web browser to render the content by setting `[browser] render_unsafe_content = yes` (see TracIni#browser-section), otherwise Trac will force the files to be downloaded as attachments for security concerns.

If the path is to a directory in the repository instead of a specific file, the source browser will be used to display the directory (identical to the result of `source:/some/dir`).

log: links

The `log:` links are used to display revision ranges. In its simplest form, it can link to the latest revisions of the specified path, but it can also support displaying an arbitrary set of revisions.

- `log:/` - the latest revisions starting at the root of the repository
- `log:/trunk/tools` - the latest revisions in `trunk/tools`
- `log:/trunk/tools@10000` - the revisions in `trunk/tools` starting from revision 10000
- `log:@20788,20791:20795` - list revision 20788 and the revisions from 20791 to 20795
- `log:/trunk/tools@20788,20791:20795` - list revision 20788 and the revisions from 20791 to 20795 which affect the given path
- `log:/tools@named-branch` - the revisions in `tools` starting from the latest revision in `named-branch` (DVCS such as Git or Mercurial)

There are short forms for revision ranges as well:

- `[20788,20791:20795]`
- `[20788,20791:20795/trunk/tools]`
- `r20791:20795` (but not `r20788,20791:20795` nor `r20791:20795/trunk`)

Finally, note that in all of the above, a revision range can be written either as `x:y` or `x-y`.

Multi-repository links

In the presence of multiple repositories, the name of the repository should be specified as the first part of the path:

- `log:repos/branch`
- `[20-40/repos]`

- `r20/repos`

See also: WikiFormatting, TracWiki, WikiPageNames, InterTrac, InterWiki