**How to use Github issues**

Each repository in Github has its own “issue tracker”. You can access it in your browser when you go to the page of the repository and click on the “Issues” tab. In the case of our repository, the issue tracker is at <https://github.com/ufal/UMR/issues>. Issues are discussions about particular problems in the repository. When the discussion reaches a solution / consensus / result, the issue can be “closed”. It is not removed, it can still be viewed, but it will not be offered in the list of issues by default. It can be reopened if it turns out that more discussion is needed. Any Github user can create an issue for our repository, including users who do not have push access to the repo. But some other actions can only be done by users who have the required permissions. New issue is created by pressing the green button “New issue”, filling out the title and the first comment for the discussion, and pressing “Submit new issue”.

If you submit an issue and then decide to rephrase a part of it, you can edit the already submitted comment. (But people will only get one e-mail notification with the first version of the comment.)

To add a new comment when viewing an issue, write the comment in the edit window, then click on the green “Comment” button. If the discussion has reached an end, you can close the issue by pressing the “Close issue” button (without comment), or enter a comment and press the same button, which now reads “Close with comment”.

The text of the issue (comment) can be formatted using MarkDown syntax; for your comfort, the edit window has a few buttons to help you with commonly used constructs, and it also has a “Preview” tab which shows what it will look like once submitted. There can be links to other issues, to files in the repository, or to any other page on the web. There can be even images (simply drag the file with the image and drop it to the edit window).

Links have the MarkDown syntax  
[text of the link](<https://the.address/folder/file>)  
When referring to a MarkDown file in the repository, you can either refer to the whole file (e.g., <https://github.com/ufal/UMR/blob/main/doc/entities.md> – simply locate the file in your browser and copy the URL), or to a particular section (e.g., <https://github.com/ufal/UMR/blob/main/doc/entities.md#taxonomy-of-entity-types> – locate the heading of the section, move your mouse to it, click on the link icon that appears to the left of the heading, the URL in the browser’s address line changes, copy it). You can also refer to a particular line or range of lines in the MarkDown source. To do that, view the file, then on the top switch from “Preview” to “Code”, then click on the line number. If you want a range of lines, press SHIFT and click on the last line of the range. Then copy the URL from the address line (e.g., <https://github.com/ufal/UMR/blob/main/doc/entities.md?plain=1#L19-L22>). Now you may ask what happens when somebody edits the file and pushes the changes to Github? The line numbers may now point to different section of the file than at the time you were creating the link. Fortunately, this is a git repository and it remembers all previous versions (commits). If you want (you do!) your link to always lead to the version that was current when writing the comment, you should use a permalink. After selecting the line range, click on the three dots that appeared next to the first line of the range, and click on “Copy permalink”. Now you have the permalink URL in your clipboard (the address link of the browser has not changed, so do not go there), and the URL is different from the above: instead of “main” (the branch name), it contains the commit identification number (e.g., <https://github.com/ufal/UMR/blob/4be8b6778551623e1c357ed9f0acd873a07c35c5/doc/entities.md?plain=1#L19-L22>).

Like in e-mails, you can quote other people by inserting “>” before each line with a quoted text. It will then be formatted differently to distinguish the quote from your own reaction. The fastest way to insert a quote is to go up in the discussion thread, locate the comment you want to quote, click on the three dots in the upper right corner of that comment, and select “Quote reply”.

You can also refer to other issues in the same repository, simply by “#” + their number; Github will resolve it to a URL that your browser understands. For example, if you put “#3” in your MarkDown source, the browser will see a link to <https://github.com/ufal/UMR/issues/3>. You can also link to individual comments in the other issues: in the initial line, e.g., “dan-zeman commented 11 hours ago”, click on “commented”, then take the new URL from the address line (e.g., <https://github.com/ufal/UMR/issues/3#issuecomment-1629717177>). Whenever you link from a comment in issue A to issue B (or to a comment in it), the page of issue B will also inform the reader that the issue has been referenced, and it will provide a back link. This works across the whole Github, so if you are in our issue tracker and reference, say, an issue in the repository of the UMR guidelines (e.g., <https://github.com/umr4nlp/umr-guidelines/issues/16>), the readers of the target issue will see that you mentioned it.

You can refer to other users by “@” + their user name. Of course, you do not have to do it, you can write “as Dan said above, …”. But if you instead write “as @dan-zeman said above, …”, Github will know that you are talking about a Github user. This may affect whether or not that user gets an e-mail notification, so you should use the “@” notation if you want the reader to pay attention to your comment and maybe reply. (Be careful. If you misspell the user name but the result is an existing user name, that user will now get notifications although they are probably not interested in our repository.) There are four levels of e-mail notifications for each user-repository pair:

* Watching All Activity: The user gets a notification for each new issue in the repository, and each new comment in existing issues in the repository, unless this user is the author of that comment. If you want to know who is watching our repository, see <https://github.com/ufal/UMR/watchers>. If you are watching the repository, you probably see an “Unwatch” button at <https://github.com/ufal/UMR> (I don’t see it but I’m the owner of the repository, so I cannot unwatch it).
* Participating and @mentions: You will get notifications for issues in which you added a comment (you are participating) or where someone mentioned you using your @user name. If you are in this state, the button where you can change it is labeled “Watch”.
* Ignore. No notifications even if mentioned using your @user name.
* Custom. You can select types of events for which you want to be notified.

The default notification behavior for a new repository to which you have push access depends on the settings in your user profile. Go to <https://github.com/settings/notifications> and see if you want to change it.

The e-mail with the notification contains the text of the new comment, although not so nicely formatted as on the Github web. There is a link that will take you to the Github page and display it in your browser. But if you just want to quickly answer in text mode, you can reply to the notification e-mail and it will post your answer as a new comment. (If you do so, you should make sure that the e-mail does not contain unnecessary quotation of the notification e-mail, long signatures etc.)