The art of a good commit message

How to write commit messages that will save you time when you look back on them in 6 months.

Signed-off-by: David Edmundson

- -o- parent 1544806b
- Branches akademy_2025
- No related tags found
- 11 1 merge request !3 The art of a good commit message

The commit message sets the tone for the whole merge request

The commit message is as important as the patch itself

The 3 parts

- Subject
- Body
- Project-specific suffixes

The subject

- Short summary (<80 characters)
- Should follow the form:
- "once merged, this commit will [....]"
 - "Fixed the build for BSD" (bad)
 - "Fix the build for BSD" (good)

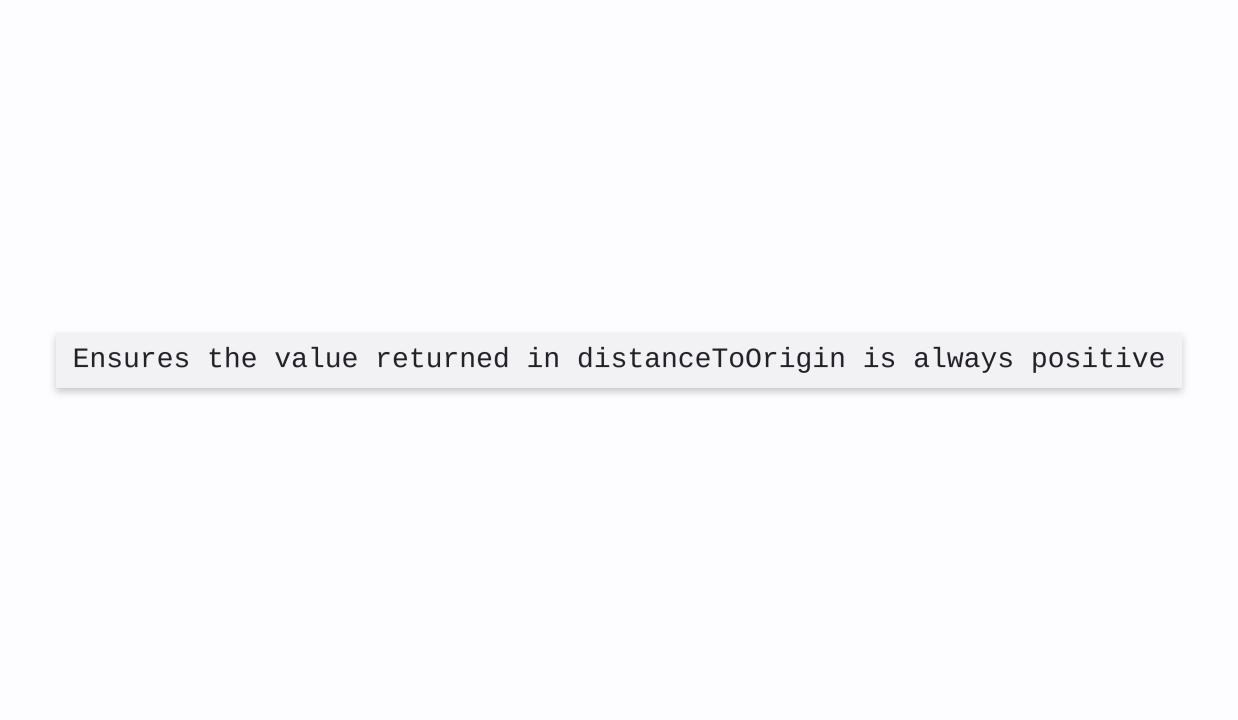
The body

The most important bit

Explain why not why

Example

```
qreal KBooglyBoo::distanceToOrigin(const QPointF &cusorPosition)
{
    qreal dx = cusorPosition.x() - m_origin.x();
    qreal dy = cusorPosition.y() - m_origin.y();
    qreal distance = std::sqrt(dx * dx + dy * dy);
    return distance;
    return std::abs(distance);
}
```



Ensures the value returned in distanceToOrigin is always positive

Absolutely Terrible

The goal of KBooglyBoo is to show an effect how far the mouse is from the hotspot corner. On right-to-left languages, m_origin is swapped, but still guaranteed to be in a corner.

Good templates

- Explain why the old code needs changing
- Explain why your new feature is benefitial to the project/user
- Explain why the code you're removing was put there to begin with (and is now obsolete)

Other

- Explain known side effects
 - New dependencies
 - Behavioural changes
- Explain where it looks like there will be a side effect but aren't

Project specific hooks

- BUG: 235234
- https://community.kde.org/Policies/Commit_Policy#
 Special_keywords_in_GIT_and_SVN_log_messages

Squash/Rebase your commits

Use a GUI

