Имена должны передавать намерения программиста

Избегать дезинформации
Когда даете имена классам или переменным, старайтесь избегать длинных, перегруженных названий. К примеру, если в одном месте у вас есть класс `XYZControllerForEfficientHandlingOfStrings`, а в другом месте — `XYZControllerForEfficientStorageOfStrings`, это слишком много лишней информации. Важно сделать имена понятными и легко находимыми в коде, но без избыточного контекста.
То есть, если ваше приложение называется «Gas Station Deluxe», не нужно добавлять к каждому классу префикс «GSD». Это только усложнит чтение кода. Лучше использовать короткие, но точные имена, которые четко описывают назначение класса или переменной, без избыточных деталей.

Функции

Первое правило: функции должны быть компактными. Второе правило: функции должны быть еще компактнее.
Максимальный уровень отступов в функции не должен превышать одного-двух.
Код должен читаться как рассказ — сверху вниз.
Выбор хорошего имени для функции способен в значительной мере объяснить смысл функции, а также порядок и смысл ее аргументов.
Например, assertEquals можно записать в виде assertExpectedEqualsActual(expected,actual). Это в значительной мере решает проблему запоминания порядка аргументов.

Классы

Система должна состоять из множества мелких классов, а не из небольшого числа больших. Каждый класс инкапсулирует одну ответственность, имеет одну причину для изменения и взаимодействует с другими классами для реализации желаемого поведения системы.

Общие рекомендации:

1. **Соблюдайте consistency** — придерживайтесь одного стиля на протяжении всего проекта.

2. **Читаемость важнее компактности** — ясные имена лучше коротких.
3. **Следите за чистотой структуры классов и файлов** — каждый класс или компонент должен быть в своём файле.
4. **Пишите комментарии** — не бойтесь оставлять пояснения для сложных участков кода.

1 Имена классов:

- Используйте **UpperCamelCase** для классов.
- Название должно четко отражать суть класса.

Пример:

```
class UserInterface;
class MainWindow;
```

2. Имена методов:

- Используйте **lowerCamelCase** для методов.
- Метод должен начинаться с глагола и чётко описывать своё действие.
- Для методов, связанных с обработкой событий, добавляйте префикс `on`, например: `onButtonClick()`.

Пример:

```
void initWindow();
void onButtonClick();
```

3. Имена полей классов:

- Используйте **lowerCamelCase** для полей класса.
- Добавляйте префикс `m_` для приватных полей.
- Поля, связанные с UI, могут иметь префикс `ui_`.

Пример:

```
class MainWindow : public QMainWindow {
    Q_OBJECT
```

```
private:
    int m_windowWidth;
    QLabel* ui_labelName;
};
```

4. Имена для UI элементов (из `.ui` файла):

- Используйте **lowerCamelCase** для названий элементов интерфейса пример: lineEditAes
- Убедитесь, что название указывает на тип элемента и его назначение.
- Избегайте аббревиатур, за исключением общепринятых (например, `btn` для кнопок).

Для именования переменных интерфейса в Qt, часто используется подход с тремя составляющими:

1. **Название UI-элемента**: описывает конкретный элемент пользовательского интерфейса (например, кнопка, текстовое поле, метка и т.д.).
2. **Название класса, к которому элемент принадлежит**: обычно совпадает с классом, который отвечает за это окно или виджет.
3. **Назначение элемента**: кратко указывает, для чего этот элемент предназначен (например, кнопка подтверждения, поле для ввода имени и т.д.).

Примеры:

1. QPushButton

- **Название UI**: `pushButton`
- **Класс**: `MainWindow`
- **Назначение**: Кнопка для сохранения данных

Итоговое имя: `pushButtonMainWindowSave`

2. QLineEdit

- **Название UI**: `lineEdit`
- **Класс**: `UserSettingsDialog`
- **Назначение**: Поле для ввода имени пользователя

Итоговое имя: `lineEditUserSettingsDialogUserName`

3. QLabel

- **Название UI**: `label`
- **Класс**: `LoginDialog`
- **Назначение**: Метка для пароля

Итоговое имя: `labelLoginDialogPassword`

Общая рекомендация:

В ui первые слова должны отображать чем является данные элемент интерфейса(lable,btn,lineEdit)
Слова которые идут после должны отображать суть элемента или с каким модулем данный эл-нт взаимодействует например:

```
lineEditAes (line edit который находится внутри AES)
btnAes256On (кнопка которая отвечает за включение aes)
```

Пример:

```
<QPushButton name="btnSubmit" />
<QLineEdit name="lineEditUsername" />
```