Часть 1 (Оформление кода)

Оформление кода

Необходимо следовать устоявшимся традициям оформления исходного кода.
Это облегчает чтение исходного кода. Кроме того, это позволяет избежать привратности в толковании исходного кода реализацией языка.
Конечно, реализация языка допускает верстку исходного кода (пробельными символами) на усмотрение программиста, но в более или менее длительной перспективе на эту лояльность со стороны реализации языка не целесообразно расчитывать.
Разумеется, в ряде случаев приходится иметь дело с отступлением от этих соглашений. Однако, целесообразно ознакомиться с соглашениями, ставшими традиционными.

О составлении имен

Имена должны характеризовать то, чему они даются (поэтому и существует столько рекомендаций о составлении имен). И при этом, они должны быть по возможности короче.
Если в именах используются аббревиатуры, то принято только первую литеру аббревиатуры прописывать заглавной.

Имена ссылочных типов

Имена классов это предпочтительно существительные. Они прописываются верблюжьей нотацией, и причем с заглавной.
Имена пакетов — как имена классов (но необходимо помнить, что в первую очередь именно эти имена должны быть уникальны). Они прописываются строчными, и при необходимости составляющие имени разделяются символом прочерка.

Имена интерфейсов это предпочтительно прилагательные (или же существительные). Они прописываются так же.
Имена перечислений — как имена классов. Они прописываются так же. А имена классов из списка перечисления прописываются заглавными литерами.
Имена обобщений — как имена классов. Они прописываются так же, как имена классов.
Имена обобщенных типов это предпочтительно не части речи, а отдельно взятые буквы. Они прописываются заглавной.
Имена аннотаций — это специфика аннотаций. И прописываются они специфично (но чаще всего с заглавной).

Имена методов

Имена методов это предпочтительно глаголы. Они прописываются верблюжьей нотацией, и причем со строчной. Они обозначают их действие.
При необходимости они включают тип возвращаемого значения, опять- таки их действие, и типы аргументов.
При необходимости составляющие их имени разделяются специальными символами. Имена методов характерного действия содержат характерные составляющие (например, у методов геттеров и сеттеров).

Имена переменных

Имена переменных это предпочтительно любая подходящая часть речи. Они прописываются верблюжьей нотацией, причем со строчной.
Имена переменных более прочих имен тяготеют к краткости. Имена переменных, используемых в глубоко вложенных структурах кода, или просто в циклах, целесообразно ограничивать единственной буквой (тогда это будет строчная).
На такой случай есть рекомендация:
для символов предпочтительны имена `c`, `d`, `e`;
для целочисленных предпочтительны имена `i`, `j`, `k`, `m`, `n`;
для fp- чисел предпочтительны относительно длинные имена;

Для переменных- констант предпочтительны относительно длинные имена. Они прописываются заглавными, а составляющие имени разделять символом подчеркивания.

Об организации файлов исходного кода

Файлы более 2000 строк считаются громоздкими, и исходный код следует хранить в менее громоздких файлах.
В файле исходного кода длину строки следует выдержать в пределах 80 символов (так как это традиционная длина строки).
Прописываемые в строках выражения следует выдерживать по возможности лаконичными (так как слишком длинные выражения могут быть сочтены человеческой ошибкой). Лаконично ли выражение — это следует определять по правилу «один- два- много». Это правило применяется к таким категориям, как глубина вложенности выражений в скобках, количество выражений в скобках на одном уровне вложенности, количество запятых в выражении, длина последовательных присваиваний, и прочее. Если же какое- либо выражение не умещается в строке, то оно разбивается по строкам. При этом надо учитывать следующее:
выражение разбивается после запятой (если она есть);
и перед оператором;
при этом, выражение, заключенное в круглые скобки не разносится по строкам;
и при этом вложенность структур кода по прежнему разобозначается отступами;

Файлы исходного кода следует оформлять в виде секвенции их секций (разделов). Секция — это просто участок исходного кода, ограниченый пустыми строками (комментариями). В наиболее общем виде файл исходного кода представляет собой секвенцию следующих секций: комментарий к файлу, импорт, и собственно исходный код.

Исходный код следует подразделять на секции. Секции заключают в себе структуры исходного кода (секциями считаются именно целые структуры, а не отдельные операторы). Структуры исходного кода располагаются в нем с соответствующими отступами. Один уровень вложенности подразумевает один отступ (величина отступа — табуляция, стандартный размер табуляции — 4 пробела).
Разные структуры исходного кода оформляются по разному.

Отдельные операторы

Отдельно взятые операторы занимают отдельные строки. Это все требования к ним.

Блочные операторы

Блочные операторы заключают в себе отдельные операторы — для отдельных операторов в блоке по прежнему действует вышеописанное правило. Сам блочный оператор оформляется так:
прописывается открывающая скобка, и переводится строка;
прописывается секвенция операторов;
за ней (с новой строки) прописывается закрывающая скобка;

Если же блок пустой (и пустой блок может потребоваться в ряде случаев — например, в цикле `for` с пустым телом), то ограничивающие его скобки прописываются просто подряд.

Операторы управления ходом вычислений

Тело этих операторов следует заключать в блок, даже если это единственный оператор. Это позволит избежать той человеческой ошибки, когда в процессе составления исходного кода единственный оператор заменяется секвенцией операторов, но программист забывает заключить ее в блок.
Сами операторы управления ходом вычислений оформляются в общем случае так:

прописывается ключевое слово, обозначающее оператор управления ходом вычислений, и через пробел прописывается выражение- условие;
в этой же строке прописывается тело оператора (по вышеописанным правилам для блочных операторов), и при этом секвенция операторов тела прописывается с отступом от ключевого слова, обозначающего оператор управления ходом вычислений, а закрывающая скобка — без отступа от него;

Прочие правила — специфика конкретных операторов управления ходом вычислений.

`while`

у этого оператора нет специфических правил оформления;

`do – while`

у этого оператора нет специфических правил оформления;
хотя, стоит заметить, что `while ( … )` прописывается в той же строке, что и закрывающая скобка тела оператора;

`for`

у этого оператора нет специфических правил оформления;

`if`

если в операторе `if` присутсвует ветвь `else`, то она прописывается в той же строке, что и закрывающая скобка тела;
причем, это касается как ключевого слова ветви, так и ее тела;
при этом, тело ветви прописывается по вышеописанным правилам для блочных операторов;
ветвь прописывается через пробел после закрывающей скобки;
при этом закрывающие скобки ветвей прописываются без отступа от верха иерархии;

если оператор `if` вложенный, то он прописывается по вышеописанным правилам для оператора `if` с ветвью `else`;

`switch`

все ветви (лучше сказать метки) этого оператора прописываются без отступа от ключевого слова, обозначающего этот оператор управления ходом вычислений;
секвенция операторов за его метками прописывается с отступом от ключевого слова, обозначающего метку;

Структуры кода для обработки прерываний

блок `try` прописывается так — ключевое слово `try` прописывается на одной строке с блоком (блок прописывается по вышеописанным правилам для блочных операторов);
причем, секвенция операторов блока прописывается с отступом от ключевого слова `try`;
блок `catch` прописывается на одной строке с закрывающей скобкой блока `try`;
причем, этот блок прописывается так же, как и блок `try`;
последующие блоки `catch` прописываются с одинаковым отступом относительно него и друг друга;
блок `finally` прописывается так же, как блок `catch`;

Методы

Методы оформляются так же, как и операторы управления ходом вычисления (в общем случае).
Разве что, вместо записи оператора управления ходом вычислений будет запись, характерная для объявления метода. Причем, в этой записи имя метода и область параметров метода принято прописывать просто подряд, без пробела.
И еще, если метод возвращает единственное значение, то в операторе возврата значения не следует использовать скобки.

Ссылочные типы оформляются так же, как и методы. Разве что, вместо записи, характерной для объявления метода, будет запись, характерная для объявления ссылочного типа.

О комментариях

Комментарии следует применять для комментирования целых файлов исходного кода, отдельных структур кода в этих файлах, и отдельных строк.
Комментарии к целым файлам должны содержать сведения касательно всего файла, комментарии к отдельным структурам кода должны содержать сведения касательно этих структур кода, комментарии к строкам кода должны содержать только сведения касательно этих строк кода.
Для комментирования файлов и структур кода в них следует применять многострочные комментарии, для комментирования отдельных строк следует применять однострочные. Секвенцию однострочных комментариев не следует применять как суррогат многострочного комментария.
Многострочный комментарий следует располагать перед комментируемым кодом, либо в конце его строки (если он настолько мал); но, многострочный комментарий не следует располагать внутри комментируемого участка исходного кода. Если один участок кода комментируется секвенцией многострочных комментариев (когда они малы, и идут в конце строки), то эти комментарии должны быть выровнены друг с другом табуляцией.
Однострочный комментарий следует располагать перед комментируемой строкой кода, либо за ней.
Отступы комментариев должны быть такие же, как у комментируемого кода.
Комментарии следует предварять пустой строкой.
Наиболее сложные для понимания участки кода следует комментировать, наиболее простые — не следует. В любом случае комментарии должны быть лаконичными.

Об употреблении пробельных символов

Об употреблении пробелов

Употребление пробелов было рассмотрено выше. Кроме того, пробелы употребляются:
после секций цикла `for;`
после запятой (где бы она ни употреблялась);
с бинарными операторами (по обе стороны от знака оператора);

Необходимо помнить, что пробелы не употребляются:
с унарными операторами (между оператором и операндом);
с оператором доступа к членам;
между именем метода и областью параметров;

Об употреблении табуляции

Употребление табуляции полностью рассмотрено выше.

Об употреблении пустых строк

Как и говорилось выше, секции файла разделяются пустыми строками. Кроме того, пустые строки употребляются и внутри секций, и ставятся так:
между вложенными в нее секциями;
между идущими подряд методами;
между объявлением переменных и следующей за ним секвенцией операторов;