02 - Meaningful Names

- Nomear é parte intrínseca do processo
- Nomes devem mostrar a intenção.
    - E alterá-los se houver um nome melhor
    - Deve dizer o quê, o como e porquê daquela variável.
    - Idealmente um nome deve dispensar comentários
- Código deve ser explícito.
    - Nomes não devem suscitar mais perguntas
- Mesmo código simples se torna difícil de entender se houver muitas informações implícitas
  NÃO DESINFORMATIVO
- Se atentar para nomes que podem ter vários significados
    - e.g. uma variável "lista" é realmente um data type lista ou só uma listagem metafórica?
        - De qualquer modo é melhor evitar data type no nome.
    - Evitar nomes longos difícil de ler à primeira vista.
    - Evitar l e O, confunde com 1 e 0.
      NÃO NÃO-INFORMATIVO
- Não basta nomes de variáveis serem diferentes, eles têm que ter significados diferentes.
    - Não é bom criar uma variável quase igual a outra só diferente por um número ou uma grafia.
    - e.g. a1 e a2. *ProductInfo* & *ProductData*
- As diferenças devem ter significado

PRONUNCIÁVEL

- É bom usar nomes pronunciáveis.
    - e.g. não usar cstmr para customer.
      BUSCÁVEL
- Importante usar nomes buscáveis
    - e.g. evitar a, i, j, k.
- Escrever nomes mais longos se necessário.
    - Nomes de uma letra só podem ser usados no máximo apenas em funções curtas em escopo local.

NÃO-CODIFICADO

- Ser explícito no que a variável faz. Evitar notação húngara.
- Evitar member prefix ("m_") para atributos de classe.
- Evitar I de interface.

- Ser claro, não usar abreviações ou inferências mentais para saber o que uma variável faz.
- Devem ser um substantivo.
    - Evitar palavras como Manager, Processor ou Info no nome.
- Métodos devem ter um verbo.

NOMES DE VARIÁVEIS

CLASSES

- Não ser fofo ou engraçadinho. Ser claro e direto.
- Usar sempre o mesmo verbo para um dado significado.
    - e.g. não misturar get com fetch e retrieve.
    - Controller & manager & driver - Tudo bem ter vários métodos em classes diferentes com o mesmo nome se conceitualmente eles fizerem a mesma coisa.
    - e.g. método *add*
    - Mas só os métodos add forem semanticamente iguais. e.g. Somar elementos e inserir um elemento na lista não podem os dois serem chamados de *add* - Tudo bem usar termos técnicos da programação - Domínio das Soluções
    - e.g. job, queue
- Quando não der, usar os termos do problema - Domínio dos Problemas
    - e.g. Customer.
- Adicionar contexto pode aumentar claridade, quando já não for claro.
    - e.g. Usar addrFirstName em vez de FirstName, quando se tratar de um endereço
    - Uso de classes pode dispensar isso se o contexto já for evidente
- Nomeação precisa ser descritiva
- Não devemos ter medo de renomear variáveis por nomes melhores.