Aula 33 — 30 Abril

- Pensar se se justifica criar um método
- O método manipula os dados da classe? Fazer uma nova classe se não manipular
  - ↳ não se justifica uma Utility por classe (usar bom senso nesta situação)
    - ↳ criar uma Utily geral para todo o projeto: UtilityProjectName
      - ↳ para evitar confundir com outra Utility
- Se uma classe é internal, tudo dentro da classe também é internal

Error Handling

- Quando validação / testes não são feitos
  - ↳ Apanhador de erros → TryCatch

Erros

- Runtime error: só é apanhado em execução, em algumas situações
- Erro de sintaxe: detetados pelo editor
- Erro de raciocínio
  - ↳ Quando der erro, comentar e usar essa versão como referência, para criar uma versão diferente
    - ↳ usar boas práticas
- Apanhador de erros é escrito no Program
  - ↳ pode tornar o código mais lento
    - ↳ validar inputs é uma alternativa
- System.Exception: classe da framework que lida com erros
- Throw; : útil na fase de development