**C# 编码格式与习惯规范**

黄天予 2022141461050

**1. 通用格式与布局**

* **a. 强制：**
  1. **缩进：** 必须使用4个空格进行缩进，禁止使用Tab字符。
  2. **大括号风格：** 必须使用ANSI风格（即左大括号与右大括号各独占一行）。
  3. **行长：** 单行代码长度不应超过120个字符。
  4. **文件编码：** 所有C#源文件必须使用 UTF-8 编码。
  5. **using 指令排序：** 必须按字母顺序排序，并分组（系统命名空间在前，第三方库次之，项目内部命名空间最后），每组之间空一行。
  6. **空白行：**
     + 类、结构体、枚举、接口的定义之间必须空两行。
     + 类内部的成员（字段、属性、方法、事件）定义之间必须空一行。
     + 方法内部的逻辑块之间可以适当添加空行，以提高可读性。
  7. **运算符两侧空格：** 二元运算符（如 +, -, \*, /, =, ==, &&, ||）两侧必须各有一个空格。
  8. **逗号和分号：** 逗号后必须有一个空格，分号前不能有空格。
  9. **括号内侧空格：** 括号内侧不应有空格。
  10. **属性访问器：** 属性的 get 和 set 访问器应在同一行或各自独立一行，取决于其复杂性。简单的自动属性应在一行。
* **b. 推荐：**
  1. **var 关键字：** 推荐在变量类型显而易见时使用 var 关键字，以提高可读性，但避免在类型不明确时使用。
  2. **表达式体成员：** 推荐对单行或非常简单的属性、方法、构造函数使用表达式体成员（=>）。
  3. **字符串插值：** 推荐使用字符串插值（$）而非 string.Format() 或字符串连接
  4. **空行：** 逻辑上相关的代码块之间可以适当添加空行，以提高可读性
  5. **单行语句：** 推荐即使是单行 if、for、while 语句也使用大括号，以避免歧义和未来扩展问题。
* **c. 允许：**
  1. **三元运算符：** 允许在简单、清晰的赋值或返回语句中使用三元运算符。
  2. **单行 if/for/while：** 在非常简单且不影响可读性的情况下，允许使用单行 if、for、while 语句（不带大括号），但需谨慎。

**2. 命名规范 (Naming Conventions)**

* **a. 强制：**
  1. **类、接口、枚举、属性、方法、事件、公共字段：** 必须使用 PascalCase。
  2. **局部变量、方法参数、私有字段：** 必须使用 camelCase。私有字段前可加下划线 \_
  3. **常量（const）：** 必须使用 PascalCase。
  4. **枚举成员：** 必须使用 PascalCase。
  5. **命名空间：** 必须使用 PascalCase，并与项目文件夹结构保持一致。
  6. **泛型类型参数：** 必须使用大写字母 T 作为前缀，如 TKey, TValue 或 TEntity。
* **b. 推荐：**
  1. **有意义的名称：** 推荐使用完整、描述性强的名称，避免使用不明确的缩写。
  2. **布尔变量/属性：** 推荐以 Is, Has, Can 等前缀命名布尔类型的变量或属性。
  3. **异步方法：** 推荐异步方法以 Async 后缀结尾（如 GetDataAsync）。
  4. **接口命名：** 推荐接口名称以大写字母 I 开头。
* **c. 允许：**
  1. **this 关键字：** 允许在需要明确区分成员变量和局部变量/参数时使用 this. 前缀，或在团队约定中统一使用。

**3. 代码结构与习惯**

* **a. 强制：**
  1. **异常捕获：** 必须捕获并处理特定的异常类型，禁止使用空的 catch (Exception) 块。
  2. **避免硬编码：** 配置信息等必须通过配置文件、环境变量等获取，禁止硬编码。
  3. **显式访问修饰符：** 类的所有成员都应显式指定访问修饰符，避免使用默认修饰符。
  4. **避免魔法数字和字符串：** 必须将代码中重复出现的、有特定含义的数字或字符串定义为具名常量或枚举。
* **b. 推荐：**
  1. **方法/函数长度：** 推荐方法体不超过50行代码，以保持其职责单一和可读性。
  2. **空值检查：** 推荐使用的空值条件运算符 (?.) 和空合并运算符 (??) 进行空值检查，提高代码简洁性。
  3. **LINQ：** 推荐在处理集合时优先使用 LINQ 表达式，以提高代码可读性和简洁性。
  4. **readonly 字段：** 推荐将只在构造函数中赋值的字段声明为 readonly。
  5. **集合初始化器：** 推荐使用集合初始化器来创建和填充集合。
  6. **对象初始化器：** 推荐使用对象初始化器来创建和初始化对象。
  7. **模式匹配：** 推荐使用的模式匹配（is 表达式、switch 表达式/语句）来简化类型检查和条件逻辑。
  8. **记录类型 (Records)：** 推荐使用的 record 类型来定义不可变的数据传输对象或值对象。
  9. **早期退出/守卫语句：** 推荐使用早期退出来处理无效参数或前置条件，避免深层嵌套的 if 语句。
  10. **目标类型 new() 表达式：** 推荐在使用目标类型new()表达式，当类型可以从上下文推断时，简化对象实例化。
  11. **init 访问器：** 推荐使用 init 访问器来创建不可变属性。
  12. **默认参数值：** 推荐在方法重载过多或参数有合理默认值时使用默认参数值，以简化方法调用。
  13. **元组：** 推荐使用元组作为轻量级的数据结构，尤其是在方法需要返回多个值时。
* **c. 允许：**
  1. **部分类：** 允许在代码生成或将一个大型类拆分为多个文件时使用部分类。
  2. **扩展方法：** 允许使用扩展方法来增强现有类型的功能，但需谨慎使用，避免滥用导致代码难以理解。
  3. **goto 语句：** 仅在极少数特定场景且能显著提高可读性时，允许使用 goto，但需谨慎。