پروژه نهایی درس طراحی سیستم‌های شیء گرا

خلاصه

هدف از این پروژه، طراحی برنامه‌ای است که از طریق آن بتوان یک پروژه‌ی نوشته شده به زبان C# را مستندسازی نمود، به نحوی که تمامی namespaceها، کلاس‌ها و سایر جزئیات پروژه در آن مستند شود و به منظور استفاده‌های بعدی، یک مستند کامل در اختیار برنامه‌نویسی (برنامه‌نویسانی) باشد که بعدها قرار است بر روی آن پروژه تغییرات حاصل نماید.

جزئیات

اطلاعاتی که بایستی از یک پروژه بتوان مستند نمود به شرح زیر هستند:

۱. اطلاعات کلی که بایستی در مورد پروژه داشته باشیم:
   - ۱.۱. شناسه‌ی پروژه (عددی است که فقط قابلیت خواندن دارد و زمان ایجاد پروژه از آن اطلاع داریم)
   - ۱.۲. نام پروژه
   - ۱.۳. نام مدیر پروژه
   - ۱.۴. تاریخ شروع و تاریخ پایان
   - ۱.۵. هدف از اجرای پروژه (توضیحات)
۲. اطلاعاتی فنی که بایستی در مورد یک پروژه داشته باشیم:
   - ۲.۱. همانطور که می‌دانید یک پروژه شامل یک یا چندین فضای نام (namespace) است.
   - ۲.۲. هر namespace خود می‌تواند شامل خصوصیاتی از قبیل زیر گردد:
     - ۲.۲.۱. یک نام دارد
     - ۲.۲.۲. صفر، یک یا چندین namespace دارد
     - ۲.۲.۳. صفر، یک یا چندین class دارد (بخش ۳)
     - ۲.۲.۴. صفر، یک یا چندین enum دارد (بخش ۴)
۳. اطلاعات فنی در مورد یک class
   - ۳.۱. یک نام دارد
   - ۳.۲. یک توضیح دارد (در مورد کاربرد این کلاس)
   - ۳.۳. سطح دسترسی دارد (بخش ۷)
   - ۳.۴. نام نویسنده‌ی این کلاس (کسی که کد اولیه این کلاس را نوشته است): این خصوصیت فقط یک بار مقداردهی می‌شود.
   - ۳.۵. نام آخرین برنامه نویس تغییر دهنده‌ی کد این کلاس
   - ۳.۶. صفر، یک یا چند خصوصیت (property) دارد (بخش ۵)
   - ۳.۷. صفر، یک یا چند تابع (function) دارد (بخش ۶)
   - ۳.۸. صفر، یک یا چند کلاس داخل آن
   - ۳.۹. صفر، یک یا چند enum داخل آن
۴. اطلاعاتی فنی در مورد enum
   - ۴.۱. یک نام دارد
   - ۴.۲. یک توضیح دارد (در مورد کاربرد این enum)
   - ۴.۳. سطح دسترسی دارد (بخش ۷)
   - ۴.۴. باید بدانیم از نوع [flag] تعریف شده یا نه

۴.۵. اعضا دارد: در مورد هر عضو بایستی نام آن، عدد انتسابی به آن و توضیح آن را نیز ذخیره کنیم. (مثال ۱)

۴.۶. نام نویسنده‌ی این enum (کسی که کد اولیه این enum را نوشته است): این خصوصیت فقط یک بار مقداردهی می‌شود.

۴.۷. نام آخرین برنامه نویس تغییر دهنده‌ی کد این enum

۵. اطلاعاتی فنی در مورد property

۵.۱. یک نام دارد

۵.۲. یک توضیح دارد (در مورد کاربرد این property)

۵.۳. سطح دسترسی دارد (بخش ۷)

۵.۴. نوع داده (datatype) دارد: datatype هر خصوصیت طبیعتا می‌تواند یکی از datatype های تعریف شده در همان پروژه باشد (classها و enumهای تعریف شده در آن پروژه) یا یکی از datatype های پیش فرض. در این پروژه، datatype های پیش فرض، موارد زیر می‌باشند:

۵.۴.۱. int, long, float, double, bool, char, string, ArrayList

۵.۴.۲. البته ممکن است یک خصوصیت به طور مثال آرایه‌ای از نوع int باشد، که این مطلب را نیز باید مد نظر داشت.

۶. اطلاعاتی فنی در مورد function

۶.۱. یک نام دارد

۶.۲. یک توضیح دارد (در مورد عملی که این تابع انجام می‌دهد)

۶.۳. سطح دسترسی دارد (بخش ۷)

۶.۴. نوع داده‌ی (datatype) خروجی دارد: در این مورد نیز نوع داده می‌تواند مانند نوع داده‌ی یک property باشد (۵.۳).

۶.۵. نام نویسنده‌ی این تابع (کسی که کد اولیه این تابع را نوشته است): این خصوصیت فقط یک بار مقداردهی می‌شود.

۶.۶. نام آخرین برنامه نویس تغییر دهنده‌ی کد این تابع

۶.۷. کد تابع دارد. یعنی متن کد آن تابع نیز قرار است ذخیره شود.

۶.۸. صفر، یک یا چند پارامتر ورودی دارد (این خصوصیت به صورت مستقیم نبایستی قابل دسترسی باشد)

۶.۹. هر پارامتر ورودی:

۶.۹.۱. نام دارد

۶.۹.۲. نوع داده دارد. مانند نوع داده‌ی یک property (۵.۳)

۶.۹.۳. نوع ارسال دارد (یکی از ۴ نوع in، out، ref یا معمولی)

۷. انواع سطوح دسترسی:

۷.۱. دقت کنید در مورد سطح دسترسی، همانطور که می‌دانید دو نوع دسته‌بندی در مورد سطح دسترسی داریم: نوع اول که Access Modifier ها هستند (internal, protected, private, public) و نوع دوم که static بودن یا نبودن را در نظر می‌گیرد و همانطور که تابحال ذکر شد، نوع دسترسی static می‌تواند در کنار هر یک از این Access Modifier ها قرار گیرد. به طور مثال یک property می‌تواند private static باشد. بنابراین طبیعتا 2×4 یعنی 8 نوع سطح دسترسی خواهیم داشت که از ترکیب static با سایر انواع دسترسی حاصل خواهد شد.

۷.۲. در مواردی که نیاز به ذخیره‌سازی سطح دسترسی دارید، از همین انواع دسترسی استفاده نمایید. (زیرا فرض می‌کنیم نوع دسترسی دیگری نداریم). بنابراین نوع دسترسی از همین ۸ مورد انتخاب خواهد شد.

۷.۳. در برخی موارد بعضی از این سطوح دسترسی غیر مجاز هستند، به طور مثال، یک enum نمی‌تواند سطح دسترسی public static داشته باشد، به این دلیل که نمی‌تواند از نوع static باشد. برای راحتی کار این محدودیت‌ها را در نظر نگیرید، اما اگر با در نظر گرفتن این محدودیت‌ها برنامه را نوشتید، نمره‌ی اضافی خواهد داشت.

مثال ۱

```
internal enum E1
{
        E11 = -4,
        E12,
        E13 = 50
}
```

همانطور که ملاحظه می‌نمایید، اعضای این enum، موارد E11، E12 و E13 هستند و اعداد انتسابی به آنها به صورت ۴-، خودکار و ۵۰ می‌باشد.

۸. خروجی‌هایی که از سیستم انتظار داریم:

۸.۱. مشاهده لیست اجزاء یک پروژه مانند namespaceها و کلاس‌ها

۸.۲. مشاهده لیست اجزاء یک namespace

۸.۳. مشاهده لیست اجزاء یک کلاس

۸.۴. مشاهده لیست برنامه‌نویسان یک پروژه (که متشکل از تمامی برنامه‌نویسانی که در نوشتن اجزاء مختلف این پروژه نقش داشته‌اند)

۸.۵. محاسبه تعداد خصوصیات و عملکردهای یک کلاس

۸.۶. یک سری عملیات بر روی توابع داخل کلاس از جمله متن کد آنها

ⓘ انجام پروژه در گروه‌های حداکثر دو نفری است.

ⓘ سعی کنید ابتدا کلاس‌های مورد نیاز را طراحی نمایید سپس فرم‌ها و interface برنامه، چرا که بخش اول مهمتر از بخش دوم خواهد بود.

ⓘ در مورد بخش ۸ نیز سعی کنید حتی الامکان ابتدا کدها و توابع مورد نیاز را طراحی نمایید.

ⓘ به منظور پیاده‌سازی ارتباط بین اجزاء مختلف می‌توانید به هر شیئ یک صفت id انتساب دهید و آن را در زیرمجموعه‌های ذکر کنید. به طور مثال به هر کلاس یک خصوصیت انتساب می‌دهیم به نام id و در کلاس propertyها، خصوصیتی به نام ClassId انتساب می‌دهیم که مقدار id کلاس مربوطه را در خود نگهداری می‌کند.

ⓘ تحویل پروژه به صورت حضوری است و حضور همه افراد گروه الزامی است.

تاریخ دقیق تحویل متعاقبا اعلام خواهد شد

موفق باشید – شکفته

۱۳۸۹/۰۳/۱۹