Всеукраїнської конференції здобувачів вищої освіти і молодих учених (17 листопада 2020 р., м. Київ). – К. : КНУТД, 2020. С.100-105.
3. Астістова Т.І., Розробка системи електронного поселення /Т.І Астістова , Д.Д.Ляховська Д.Д./ Тези доповідей V Міжнародна науково-практична конференція «Мехатронні системи: інновації та інжиніринг» - «MSIE-21», КНУТД 4 листопада 2021, С. 150- 152.

АСТІСТОВА Т.І., МИРЕЦЬ Р. В.

ВИКОРИСТАННЯ ПАТЕРНІВ ПРОЕКТУВАННЯ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ASTISTOVA T.I., MYRETS R. V .

USING DESIGN PATTERNS FOR SOFTWARE DEVELOPMENT

Annotation. An effective way to solve problems in software design is to use software templates. It is known that the template is not a finished sample that we can translate into program code.

Programming patterns are developed effective approaches, techniques and rules for solving problems when creating softwareю.Considered the difference between each of the patterns, the complexity implementation of each and interaction between components. Found pros and cons each.

Keywords: software, interface, patterns, Singletone, algorithm.

Вступ

На даному етапі розвитку комп'ютерної техніки програміст повинен використовувати сучасні технології розробки, щоб створити ефективну, корисну програму, яка буде ще й прибутковою. Процес руху в розробці веб-додатків до вдосконалості, буде вважатися завершеним тоді, коли інтерфейс і поведінка програми не буде залежити від того, де вони виконуються на локальному комп'ютері або на віддаленому сервері, поставляючи результати своєї роботи через мережу.

Ще одна складність, з якою постійно стикаються розробники веб-додатків, - це стандартизація. Зараз необхідно, щоб всі, що є через Інтернет, було перевірено на сумісність роботи з браузерами , щоб гарантувати, що всі відвідувачі зможуть отримати максимум користі з вашого сайту.

Постійний поштовх до спрощення проектування клієнт-серверних додатків, створив низку систем та рішень, що відрізняються між собою функціональністю, архітектурою, набором дизайнерських рішень.

У світі постійно хтось стикається з такими ж проблемами проектування. Багато розробників вирішують абсолютно ідентичні завдання і знаходять схожі рішення. Щоб не створювати те, що вже створено, доречно використовувати готові шаблони (патерни) проектування.

Основна частина

Патерни з'явилися саме тому, що багато розробників шукали шляхи підвищення гнучкості і ступеня повторного використання своїх програм.

Патерни придумали та сформували в першу чергу для того, щоб пришвидшити реалізацію певних стандартизованих задач згідно даних шаблонів-заготовок, що в свою чергу полегшить життя програмісту. А не кожного разу придумувати колесо.

Патерни — це не тільки рецепт побудови коду певним чином, але й опис проблем, які призвели до такого рішення.

Ефектним способом вирішення задач при проектуванні програмного забезпечення є використання шаблонів програмного забезпечення. Відомо,що шаблон не є закінченим зразком, що ми можемо транслювати в програмний код.

Патерни програмування – це напрацьовані ефективні підходи, техніки та правила вирішення задач при створенні програмного забезпечення. Вони не прив'язуються до певної мови програмування і можуть бути застосованими в основному незалежно від конкретної мови.

Є багато різноманітних категорій і видів патернів від різноманітних програмістів. Але ми дамо класифікацію найбільш поширених патернів, які групуються у категорії патернів, їх види та складові**.**

Використання архітектурних патернів для проектування клієнт-серверних додатків спростить основні операції при проектуванні таких додатків. Це буде мати досить добрі перспективи в майбутньому.

Шаблони або патерни проектування – це керівництво для рішенням проблем, що повторюються. Це не класи, пакети або бібліотеки, які можна було б підключити до вашого застосування. Це є методикою, яка вирішує певні проблеми в певних ситуаціях.

Шаблон проєктування, чи патерн, у розробці програмного забезпечення – повторювана архітектурна конструкція, що є вирішенням проблеми проектування, у рамках деякого часто виникаючого контексту (Вікіпедія).

Слід зауважити, що шаблони проєктування не є рішенням усіх проблем, тому не завжди потрібно їх використовувати. Шаблони – це підходи до рішення проблем, і якщо їх правильно використати в потрібних місцях, то вони можуть стати порятунком. Патерн являє собою не якийсь конкретний код, а загальний принцип вирішення певної проблеми, який майже завжди треба підлаштовувати для потреб тієї чи іншої програми.

Патерни часто ототожнюють з алгоритмами. Обидва поняття описують типові рішення відомих проблем. Але алгоритм — це чіткий набір дій, а патерн — це високорівневий опис рішення, реалізація якого може відрізнятися у двох різних програмах.

Кожен патерн вирішує такі пункти:

- проблема, яку вирішує патерн;

- мотивація щодо вирішення проблеми способом, який пропонує патерн;
- структура класів, складових рішення;
- приклад однією з мов програмування;
- особливості реалізації в різних контекстах;
- зв'язки з іншими патернами.

Патерни проєктування описують типові способи вирішення поширених проблем при проєктуванні програм.

1. Користь патернів:

- перевірені рішення. Використовуючи готові рішення, ви витрачаєте менше часу;
- стандартизація коду. Використовуючи типові уніфіковані рішення, ви робите менше прорахунків при проєктуванні;
- загальний словник програмістів. Ви вимовляєте назву патерна, інші програмісти, розуміють, які класи ви використовували або який був дизайн для цього потрібні.

Найбільш універсальні — архітектурні патерни, які можна реалізувати практично будь-якою мовою. Вони потрібні для проєктування всієї програми, а не окремих її елементів.

За призначенням, виділяють наступні три основні групи патернів:

- Породжуючі патерни - піклуються про гнучке створення об'єктів без внесення в програму зайвих залежностей, надають рекомендації та техніки для створення нових об'єктів. Існує 5 породжуючих патернів.
- Структурні патерни - показують різні способи побудови зв'язків між об'єктами. Всього є 7 структурних патернів.
- Поведінкові патерни - піклуються про ефективну комунікацію між об'єктами, надають рекомендації для реалізації тої чи іншої поведінки-функції існуючого об'єкта. Є 11 поведінкових патернів. Це шаблони, які призначені для створення екземпляра об'єкту або групи пов'язаних об'єктів.

Один із подорожуючих патернів Singletone – Одинак. На сьогодні це найпоширеніший патерн серед розробників програмних додатків. Він вирішує відразу дві проблеми: а) Гарантує наявність єдиного екземпляра класу (наприклад, бази даних); б) Глобальний доступ до одного об'єкта.

Дуже вразливий приклад Одинака з нашого життя. У державі може бути тільки один офіційний Уряд держави. Незалежно від того, хто конкретно засідає в уряді, він має глобальну точку доступу до «Уряд країни N».

З розвитком реактивного програмування все більше набувають популярності патерни, в основу яких лежить патерн Спостерігач. Одним з таких є MVVM. Суть цього патерна полягає в тому, що компоненти не взаємодіють між собою на пряму, а реагують на зміну ViewModel. Отже,

MVVM має знайомі нам компоненти (Model, View) та новий (ViewModel) (Модель уявлення).

Висновки

Було досліджено та проаналізувано найбільш поширені види патернів.Кожен з них краще використовувати в той чи інщій ситуації. Немає єдиного вірного патерна .

При розробці клієнт-серверного додатку важливо вибрати патерн, який повністю задовільнить всі потреби розробників та самого продукта . Під час розробки клієнт-серверного додатку найчастіше використовують такі шаблони як Adapter, Observer та Singleton.

Література

1. Електронний ресурс. – Режим доступу: http://citforum.ua / Портал аналітичної інформації в галузі інформаційних технологій CitForum
2. Інформаційно-комунікаційні технології. Веб - сайт ООН [Електронний ресурс]. – Режим доступу: http://www.un.org/ru/development/ict/index.shtml
3. Електронний ресурс - Режим доступу: http://www.pil-network.com/#uk// Microsoft Partners in Learning

АСТІСТОВА Т. І., МНОЖИНСЬКИЙ Б. Г.

РОЗРОБКА ІНФОРМАЙІЦНО-ПОШУКОВОЇ СИСТЕМИ ДЛЯ АВТОМАТИЗОВАНОГО ЗБОРУ ДАНИХ

ASTISTOVA T. I., MNOZHYNSKY B. H.

DEVELOPMENT OF INFORMATION AND SEARCH SYSTEM FOR AUTOMATED DATA COLLECTION

Annotation. The relevance of the task related to the creation of search engines is determined by a number of factors:

- in connection with the growing amount of information, new search tools are needed;

- it is necessary to improve the quality (relevance) of search information.

It is known that there is a significant inconsistency between the information needs of a specific user and their expression in the form of information search requests prepared for one or another information system

The task of the work is the development of the structure of a new search system for the Internet and the development of information search algorithms with the determination of the relevance of found documents

Keywords: Sites, algorithm, earch engines, relevance, searching system , DBMS, document indexing,

Вступ

Актуальність задачі, пов'язаної із створенням пошукових систем визначається низкою факторів:

- у зв'язку з зростаючим обсягом інформації потрібні нові пошукові