**Курсова работа**

От: Мариян Неделев 12б №22

По: Софтуерна документация

Тема: Софтуерна документация

I.Теоретична част

1. ***Софтуерна документация***

**Софтуерна документация** е написан текст или илюстрация, която придружава компютърен софтуер или е вградена в изходния код. Документацията обяснява как функционира софтуерът или как да се използва и може да означава различни неща за хората в различни роли. Документацията е важна част от софтуерното инженерство.

**Видовете документация включват:**

* **Изисквания** - Изявления, които идентифицират атрибути, възможности, характеристики или качества на системата.
* **Архитектура / Дизайн** - Преглед на софтуера. Включва връзки с околната среда и принципи, които да се използват при проектирането на софтуерни компоненти.
* **Технически** - Документиране на код, алгоритми, интерфейси и API.
* **Краен потребител** - Ръководства за крайния потребител, системните администратори и помощния персонал.
* **Маркетинг** - Как да продаваме продукта и анализ на пазарното търсене.

1. ***Документация за изискванията***

**Документацията за изискванията** е описанието на това, което даден софтуер прави или трябва да направи. Използва се по време на разработката. Изискванията се използват от всички, участващи в производството на софтуер: крайни потребители, клиенти, ръководители на проекти, продажби, маркетинг, софтуерни архитекти, дизайнери, разработчици и тестери. Могат да бъдат целеви, близо до дизайна. Те могат да бъдат посочени като твърдения на естествен език, като изчертани фигури, като подробни математически формули и като комбинация от всички тях. Необходимостта от документация за изискванията обикновено е свързана със сложността на продукта, въздействието на продукта и продължителността на живота на софтуера.

* Ако софтуерът е много сложен или разработен от много хора, изискванията могат да помогнат за по-добрата комуникация какво да се постигне. Ако се очаква софтуерът да работи само месец или два, може да е необходима много ограничена документация за изискванията.
* Ако софтуерът е от критично значение за безопасността и може да има отрицателно въздействие върху човешкия живот, често се изисква по-официална документация за изискванията.
* Ако софтуерът е първа версия, която е надградена по-късно, документацията за изискванията е много полезна, когато управлявате промяната на софтуера и проверявате, че нищо не е нарушено в софтуера, когато е модифициран.

1. ***Архитектурна проектна документация***

**Архитектурната документация** (известна още като описание на софтуерната архитектура) е специален тип проектни документи. Много важна част от документа за проектиране при разработването на софтуер за предприятието е документът за проектиране на база данни. Той съдържа концептуални, логически и физически елементи на дизайна. Включва официалната информация, от която се нуждаят хората, които взаимодействат с базата данни. Целта на подготовката му е да се създаде общ източник, който да се използва от всички.

**Потенциалните потребители са:**

* Дизайнер на база данни
* Разработчик на база данни
* Администратор на база данни
* Дизайнер на приложения
* Разработчик на приложения

**Когато говорим за релационни бази данни, документът трябва да включва следната информация:**

* Таблици,
* атрибути и техните свойства,
* изгледи,
* ограничения като първични ключове, външни ключове.  
  Също така е много важно да актуализирате документите, тъй като всяка промяна се случва и в базата данни.

1. ***Техническа документация***

Важно е кодовите документи, свързани с изходния код (които могат да включват README файлове и API документация), да бъдат задълбочени, но не толкова многословни, че да отнемат твърде много време или да се поддържат трудно. Обикновено се намират различни ръководства за документация, специфични за софтуерното приложение или софтуерния продукт, документирани от API авторите. Тази документация може да се използва от разработчици, тестери, а също и крайни потребители. Кодовите документи често са организирани в справочно ръководство - стил, позволяващ на програмист бързо да търси произволна функция или клас.

1. **Техническа документация, вградена в изходния код;**

Често инструменти като Doxygen, NDoc, Visual Expert, Javadoc, JSDoc, EiffelStudio, Sandcastle, ROBODoc, POD, TwinText или Universal Report могат да се използват за автоматично генериране на кодовите документи - тоест те извличат коментарите , където е възможно, от изходния код и създават справочни ръководства като текстови или HTML файлове. Идеята за автоматично генериране на документация е привлекателна за програмистите по различни причини. Например, тъй като е извлечен от самия изходен код (например чрез коментари), програмистът може да го напише, докато се позовава на кода, и да използва същите инструменти, използвани за създаване на изходния код, за да направи документацията. Това значително улеснява поддържането на документацията актуална. Разбира се, недостатъкът е, че само програмистите могат да редактират този вид документация и от тях зависи да я опресняват.

1. **Грамотно програмиране;**

Уважаваният компютърен учен Доналд Кнут отбеляза, че документацията може да бъде много труден процес на последващо обмисляне и се застъпи за грамотно програмиране, написано едновременно и на място като изходния код и извлечено с автоматични средства. Езиците за програмиране Haskell и CoffeeScript имат вградена поддръжка за проста форма на грамотно програмиране, но тази поддръжка не се използва широко.

1. **Изяснително програмиране;**

То е резултат от практически приложения на грамотното програмиране в реални контексти на програмиране. Парадигмата Elucidative предлага изходният код и документацията да се съхраняват отделно. Често разработчиците на софтуер трябва да могат да създават и да имат достъп до информация, която няма да бъде част от самия изходен файл. Такива пояснения обикновено са част от няколко дейности за разработка на софтуер, където изходният код на трета страна се анализира по функционален начин.

1. ***Потребителска документация***

**Потребителските документи** просто описват как се използва дадена програма. Обикновено потребителската документация описва всяка функция на програмата и подпомага потребителя при реализирането на тези функции. Много е важно документите на потребителя да не са объркващи и те да са актуални. Последователността и простотата са много ценни. Потребителската документация се счита за договор, указващ какво ще прави софтуерът. API Writers имат добри постижения в писането на потребителски документи. Потребителската документация има три основни начина, по които потребителската документация може да бъде организирана.

* **Урок:** Учебният подход се счита за най-полезен за нов потребител, в който той се ръководи през всяка стъпка от изпълнението на определени задачи.
* **Тематични:** Тематичен подход, при който главите или разделите се концентрират върху една конкретна област на интерес, е от по-общо приложение за междинен потребител. Този подход обикновено се практикува от динамична индустрия, като информационните технологии, където потребностите на потребителите до голяма степен са свързани с изискванията за отстраняване на неизправности.
* **Списък или справка:** Този последен подход е по-полезен за напреднали потребители, които знаят точно какъв вид информация търсят.

**В случай на потребителска документация, процесът, както обикновено се случва в индустрията, се състои от пет стъпки:**

* Потребителски анализ, основната изследователска фаза на процеса.
* Планиране или действителната фаза на документиране.
* Проект за преглед, самообяснителна фаза, при която се търси обратна връзка по проекта, съставен в предишната стъпка.
* Тестване на използваемост, при което използваемостта на документа се тества емпирично.
* Редактиране, последната стъпка, в която информацията, събрана в стъпки трета и четвърта, се използва за създаване на окончателния проект.

1. ***Потребителска документация***

За много приложения е необходимо да има някои рекламни материали, които да насърчават случайните наблюдатели да отделят повече време за изучаване на продукта.

**Тази форма на документация има три цели:**

* Да развълнува потенциалния потребител за продукта и да му внуши желание да се ангажират по-активно с него.
* Да ги информира какво точно прави продуктът, така че очакванията им да са в съответствие с това, което ще получат.
* Да се ​​обясни позицията на този продукт по отношение на други алтернативи.

I.Практична част

Условие: Да се състави формуляр на html документ, който изисква да попълниш име, фамилия, клас и да избереш кой е любимия ти предмет, който изучаваш с най-голямо желание, също и кой с най-малко желание.

<html>

    <head>

        <title>www.Страница на ПГЕЕ</title>

    </head>

    <body>

        <form> <!--Задавам начало на формуляра-->

            Име:<input type="text" maxlength="20"></input><br> <!--Задавам първи текстбокс с контролата "TextBox" и максимален брой символи 20-->

            Фамилия:<input type="text" maxlength="20"></input><br> <!--Задавам втори текстбокс с контролата "TextBox" и максимален брой символи 20-->

            клас:<input type="text" maxlength="100"></input><br> <!--Задавам трети текстбокс с контролата "TextBox" и максимален брой символи 100-->

        </form> <!--Задавам край на формуляра-->

        <form> <!--Задавам начало на формуляра-->

            <p>Кой предмет изучавате с най-голямо желание:<br>

                <input type="checkbox">Математика</input><br> <!--Задавам първи чекбокс с контролата CheckBox-->

                <input type="checkbox">Български език</input><br> <!--Задавам втори чекбокс с контролата CheckBox-->

                <input type="checkbox">Физика</input><br> <!--Задавам трети чекбокс с контролата CheckBox-->

                <input type="checkbox">Интернет програмиране</input><br> <!--Задавам четвърти чекбокс с контролата CheckBox-->

                <input type="checkbox">ФВС</input><br> <!--Задавам пети чекбокс с контролата CheckBox-->

                <input type="checkbox">Биология</input><br> <!--Задавам шести чекбокс с контролата CheckBox-->

                <input type="checkbox">Английски език</input><br> <!--Задавам седми чекбокс с контролата CheckBox-->

                <input type="checkbox">Химия</input><br> <!--Задавам осми чекбокс с контролата CheckBox-->

            </p>

        </form> <!--Задавам край на формуляра-->

        <form> <!--Задавам начало на формуляра-->

            <p>Кой предмет изучавате с най-малко желание:

                <select> <!--Задавам начало на менюто с контролата Menu-->

                    <option>Математика</option> <!--Задавам първи елемент от менюто-->

                    <option> Български език </option> <!--Задавам втори елемент от менюто-->

                    <option> Физика </option> <!--Задавам трети елемент от менюто-->

                    <option> Интернет програмиране </option> <!--Задавам четвърти елемент от менюто-->

                    <option> ФВС </option> <!--Задавам пети елемент от менюто-->

                    <option>Биология</option> <!--Задавам шести елемент от менюто-->

                    <option> Английски език </option> <!--Задавам седми елемент от менюто-->

                    <option> Химия </option> <!--Задавам осми елемент от менюто-->

                </select> <!--Задавам край на менюто-->

            </p>

        </form> <!--Задавам край на формуляра-->

        <form> <!--Задавам начало на формуляра-->

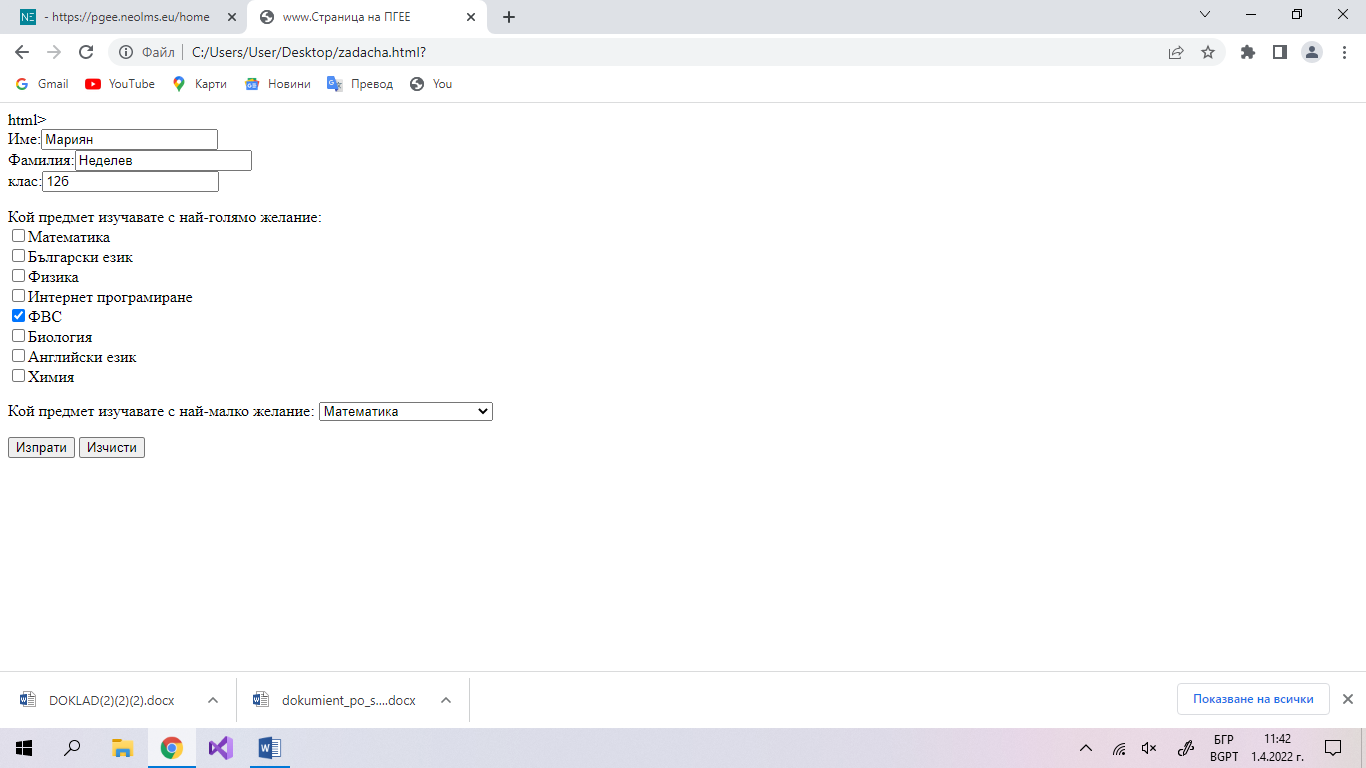
            <button type = "submit">Изпрати</button> <!--Задавам първи бутон с атрибута button към контрола Submit-->

            <button type = "submit">Изчисти</button> <!--Задавам втори бутон с атрибута button към контрола Submit-->

        </form> <!--Задавам край на формуляра-->

    </body>

</html>



Софтуерната документация е направена с коментари след всеки ред от кода за улеснение на потребителите. . Ако се използва този код трябва да се „copy-paste”- не в notepad и да се запише с разширение .html и след това може да се отваря резултата от този код.