**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ГАБРОВО**

Катедра „Компютърни системи и технологии“

**КУРСОВА РАБОТА**

**Тема**: Напишете HTML5 страница, в която се визуализира текущото време във формат „час:минути“. Информацията за времето се получава от Web ресурс getCurrentTime чрез използване на AJAX заявки.

**Разработил**: Богомил Георгиев Иванов

**Фак. №**: 22372126 **Курс**: 2-ри курс **Специалност**: СКИ

**Проверил**:.......................................

**Подпис**:........................... **Дата**:........................

**Теоретична част:**

**Асинхронно и синхронно изпълнение на програмен код**

*Синхронно изпълнение*

Синхронното изпълнение означава, че задачите се изпълняват една след друга. Всяка следваща задача започва, след като предходната завърши. Това е стандартният начин на изпълнение на код в повечето програмни езици.

**Предимства:**

* Лесен за разбиране и отстраняване на грешки код.
* Логическата последователност на задачите е ясна.

**Недостатъци:**

* Ако една задача отнема много време (напр. мрежова заявка), програмата замръзва, докато задачата се изпълни.
* Намалена производителност, особено при обработка на множество задачи.

*Асинхронно изпълнение*

Асинхронното изпълнение позволява на програмата да изпълнява задачи, без да изчаква завършването на други. Това е особено полезно за операции с вход/изход, мрежови заявки и таймери.

**Предимства:**

* По-висока производителност, тъй като програмата може да обработва други задачи, докато чака за резултати от бавни операции.
* Подходящо за потребителски интерфейси, където блокирането е неприемливо.

**Недостатъци:**

* По-сложен за разбиране и поддръжка код.
* Управлението на зависимости между задачите изисква специални техники (напр. callback функции, промиси, async/await).

**Asynchronous JavaScript and XML (AJAX). Генериране на AJAX заявки чрез библиотеката jQuery**

AJAX е техника, която позволява на уеб страници да обменят данни със сървъра асинхронно, без да презареждат цялата страница. Това подобрява потребителското изживяване, като осигурява бърза и динамична реакция на уеб приложенията.

**Примерни приложения на AJAX:**

* Автоматично обновяване на съдържание (напр. времето, новини, курсове на валути).
* Динамично зареждане на части от страница (напр. зареждане на допълнителни коментари или продукти).
* Проверка на потребителски данни (напр. уникалност на имейл при регистрация).

**Предимства на използване на jQuery за AJAX**

* Оптимизиран и лесен за използване синтаксис.
* Поддържа различни браузъри, като се грижи за несъвместимостите.
* Предлага удобни методи като $.ajax(), $.get(), и $.post() за различни типове заявки.

**Програмна реализация:**

* *index.html*

<!DOCTYPE html>

<html lang="bg">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"> </script>

<style>

h1 {

text-align: center;

font-family: "Poppins", sans-serif;

font-weight: 700;

font-style: normal;

}

#timeDisplay, #dateDisplay {

text-align: center;

font-family: "Poppins", sans-serif;

font-weight: 500;

font-style: normal;

}

</style>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&family=Rubik+Mono+One&display=swap" rel="stylesheet">

<title>Current Time</title>

<script src="getTime.js"></script>

</head>

<body>

<h1>Current Time</h1>

<h2 id="timeDisplay">Fetching time...</h2>

<h2 id="dateDisplay">Fetching date...</h2>

</body>

</html>

* *getTime.js*

$(document).ready(function() {

function fetchTime() {

$.ajax({

url: "http://kst.tugab.bg/pis/gettimejson.php",

method: "GET",

dataType: "json",

success: function(data) {

var fullValue = data.value.split(", ");

var time = fullValue[0];

$("#timeDisplay").text("Time: " + time);

},

error: function(error) {

console.error("Error fetching time:", error);

}

});

}

function fetchDate() {

$.ajax({

url: "http://kst.tugab.bg/pis/gettimejson.php",

method: "GET",

dataType: "json",

success: function(data) {

var fullValue = data.value.split(", ");

var date = fullValue[1];

$("#dateDisplay").text("Date: " + date);

},

error: function(error) {

console.error("Error fetching date:", error);

}

});

}

fetchTime();

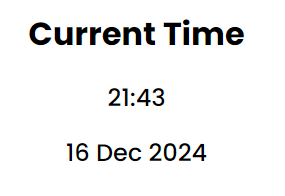
fetchDate();

setInterval(fetchTime, 1000);

setInterval(fetchDate, 86400000);

});

**Резултат от програмния код:**



**Изводи:**

Използването на AJAX позволява по-добро потребителско изживяване чрез асинхронно зареждане на данни. Библиотеката jQuery значително опростява работата с AJAX заявки, като осигурява по-кратък и разбираем синтаксис. Основен недостатък на AJAX приложенията е зависимостта от стабилността и наличността на мрежовата връзка.

**Използвана литература:**

1. Mozilla Developer Network (MDN): AJAX Introduction
2. jQuery Documentation: jQuery.ajax()
3. David Flanagan, "JavaScript: The Definitive Guide", O'Reilly Media.