



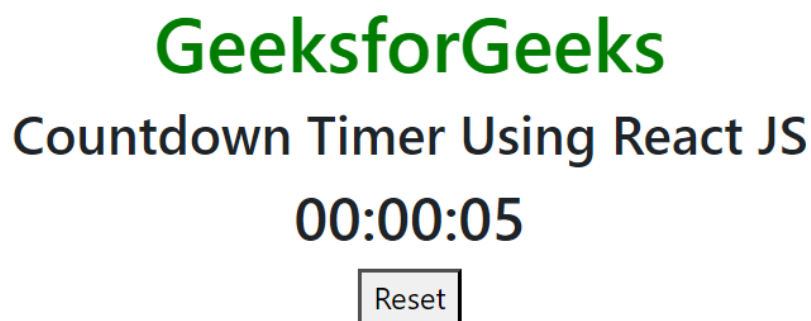
How to Create Countdown Timer in React JS

Last Updated : 04 Mar, 2024

React timers are very common **UI components** that are widely used in various applications and websites to visually display the remaining time for a specific activity or event. React timers are mostly used to highlight the **commencement** or **conclusion of events** or **offers** on commercial websites.

This tutorial explores the process of **creating a countdown timer** using **React JS**, a popular JavaScript user interface toolkit.

Preview of final output:



Предварительный просмотр выходных данных

Предварительные требования

- [NPM & Node.js](#)
- [React JS](#)
- [Хуки React JS](#)
- [Метод setInterval в JavaScript](#)

Преимущества таймера обратного отсчета

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

Понятно!

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

Здесь у нас есть 2 подхода к созданию таймера обратного отсчета, приведенные ниже

1. Использование [хуков React](#) с [функциональными компонентами](#)
2. Используя [React State](#) с [компонентами на основе классов](#)

Шаги по созданию таймера обратного отсчета с помощью React JS

Прежде чем создавать таймер в [React JS](#), вы должны быть знакомы со следующими терминологиями:

- **Сохранение времени:** Вычислите разницу между целевым таймером и текущим временем, которое у нас есть. Эта функция проверит время, оставшееся от целевого таймера, путем вычислений и вернет общее количество часов, минут и секунд.
- **startTimer:** Эта функция начнет отсчет времени после получения общего количества часов, минут и секунд из функции getTimeRemaining.
- **ClearTimer:** Эта функция используется для сброса таймера, что означает, что при перезапуске таймера очищается время, оставшееся от предыдущего обратного отсчета, в противном случае он начинает параллельный двойной отсчет или они могут сворачиваться друг с другом.
- **getDeadTimer:** This function provides the deadline of the timer means it gives time from where you want to start the countdown. In this, you have to add time if you want to extend. We have used this in two scenarios first when the page is loaded and second when someone clicks the reset button.

Шаги по созданию приложения React

Сначала давайте создадим образец веб-приложения React, на которое мы установим таймер React.

Шаг 1: Создайте приложение React с помощью следующей команды:

```
npx create-react-app foldername
```

Шаг 2: После создания папки проекта, перейдите в папку проекта

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

```
cd foldername
```

Подход 1: создайте таймер обратного отсчета с помощью React Hooks

Мы можем создать таймер обратного отсчета, используя [перехватчики React](#) и [метод setInterval](#) JavaScript.

Мы используем хуки React, такие как **useState**, **useRef** и **useEffect**

Мы предоставили рабочий код, чтобы должным образом продемонстрировать, как создать таймер обратного отсчета, используя перехватчики React с функциональными компонентами.

Таймер обратного отсчета React с использованием React Hooks Пример

В этом примере реализован таймер обратного отсчета в React с использованием перехватчиков React и метода setInterval() JavaScript.

Javascript

```
// Filename - App.js

import React, { useState, useRef, useEffect } from "react";

const App = () => {
  // We need ref in this, because we are dealing
  // with JS setInterval to keep track of it and
  // stop it when needed
  const Ref = useRef(null);

  // The state for our timer
  const [timer, setTimer] = useState("00:00:00");

  const getTimeRemaining = (e) => {
    const total =
      Date.parse(e) - Date.parse(new Date());
    const seconds = Math.floor((total / 1000) % 60);
    const minutes = Math.floor(
      (total / 1000 / 60) % 60
    );
    const hours = Math.floor(
      (total / 1000 / 60 / 60) % 24
    );
  };
};
```

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

```

        minutes,
        seconds,
    };
};

const startTimer = (e) => {
    let { total, hours, minutes, seconds } =
        getTimeRemaining(e);
    if (total >= 0) {
        // update the timer
        // check if less than 10 then we need to
        // add '0' at the beginning of the variable
        setTimer(
            (hours > 9 ? hours : "0" + hours) +
            ":" +
            (minutes > 9
                ? minutes
                : "0" + minutes) +
            ":" +
            (seconds > 9 ? seconds : "0" + seconds)
        );
    }
};

const clearTimer = (e) => {
    // If you adjust it you should also need to
    // adjust the Endtime formula we are about
    // to code next
    setTimer("00:00:10");

    // If you try to remove this line the
    // updating of timer Variable will be
    // after 1000ms or 1sec
    if (Ref.current) clearInterval(Ref.current);
    const id = setInterval(() => {
        startTimer(e);
    }, 1000);
    Ref.current = id;
};

const getDeadTime = () => {
    let deadline = new Date();

    // This is where you need to adjust if
    // you intend to add more time
    deadline.setSeconds(deadline.getSeconds() + 10);
    return deadline;
};

// We can use useEffect so that when the component

```

```

useEffect(() => {
  clearInterval(getDeadTime());
}, []);

// Another way to call the clearInterval() to start
// the countdown is via action event from the
// button first we create function to be called
// by the button
const onClickReset = () => {
  clearInterval(getDeadTime());
};

return (
  <div
    style={{ textAlign: "center", margin: "auto" }}>
    <h1 style={{ color: "green" }}>
      GeeksforGeeks
    </h1>
    <h3>Countdown Timer Using React JS</h3>
    <h2>{timer}</h2>
    <button onClick={onClickReset}>Reset</button>
  </div>
);
};

export default App;

```

Steps To Run Application: Run the application using the following command from the root directory of the project:

```
npm start
```

Output: Now open your browser and go to <http://localhost:3000>

GeeksforGeeks

Countdown Timer Using React JS

00:00:09

Reset

- **useEffect Hook** starts the timer by calling **clearTimer**.
- **getTimeRemaining Function** calculates the remaining time from a future deadline
- **startTimer Function** updates the timer display every second based on the remaining time.
- **clearTimer Function** resets the timer to 10 seconds and starts the countdown.
- **getDeadTime Function** calculates the deadline time for the countdown.
- **onClickReset Function** resets the timer to 10 seconds when the “Reset” button is clicked.
- **Return JSX** displays the timer value and a “Reset” button in the UI.

Approach 2: Create Countdown Timer Using React State in Class Components

We can create a countdown timer using [React states](#) in class components and the [setInterval method](#) of JavaScript.

We will be using the [React setState method](#) and [createRef](#) methods to update the timer in the state variable.

We have provided the working code to properly demonstrate how to create a countdown timer using the React States with Class Components.

Countdown timer using React States Example:

Javascript

```
// Filename - App.js

import React, { Component } from "react";

class App extends Component {
  constructor(props) {
    super(props);

    // We need ref in this, because we are dealing
    // with JS setInterval to keep track of it and
    // stop it when needed
    this.Ref = React.createRef();

    // The state for our timer
```

```

getTimeRemaining = (e) => {
  const total = Date.parse(e) - Date.parse(new Date());
  const seconds = Math.floor((total / 1000) % 60);
  const minutes = Math.floor((total / 1000 / 60) % 60);
  const hours = Math.floor((total / 1000 / 60 / 60) % 24);
  return {
    total,
    hours,
    minutes,
    seconds,
  };
};

startTimer = (e) => {
  let { total, hours, minutes, seconds } = this.getTimeRemaining(e)
  if (total >= 0) {
    // update the timer
    // check if less than 10 then we need to
    // add '0' at the beginning of the variable
    this.setState({
      timer:
        (hours > 9 ? hours : "0" + hours) +
        ":" +
        (minutes > 9 ? minutes : "0" + minutes) +
        ":" +
        (seconds > 9 ? seconds : "0" + seconds),
    });
  }
};

clearTimer = (e) => {
  // If you adjust it you should also need to
  // adjust the Endtime formula we are about
  // to code next
  this.setState({ timer: "00:00:10" });

  // If you try to remove this line the
  // updating of timer Variable will be
  // after 1000ms or 1sec
  if (this.Ref.current) clearInterval(this.Ref.current);
  const id = setInterval(() => {
    this.startTimer(e);
  }, 1000);
  this.Ref.current = id;
};

getDeadTime = () => {
  let deadline = new Date();

```

```

};

// We can use componentDidMount so that when the component
// mount the timer will start as soon as possible
componentDidMount() {
  this.clearTimer(this.getDeadTime());
}

// Another way to call the clearTimer() to start
// the countdown is via action event from the
// button first we create function to be called
// by the button
onClickReset = () => {
  this.clearTimer(this.getDeadTime());
};

render() {
  return (
    <div style={{ textAlign: "center", margin: "auto" }}>
      <h1 style={{ color: "green" }}>GeeksforGeeks</h1>
      <h3>Countdown Timer Using React JS</h3>
      <h2>{this.state.timer}</h2>
      <button onClick={this.onClickReset}>Reset</button>
    </div>
  );
}
}

export default App;

```

Output: Now open your browser and go to <http://localhost:3000>

GeeksforGeeks

Countdown Timer Using React JS

00:00:09

Reset

Explanation:

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

- In `componentDidMount` initiate the countdown by calling the `clearTimer` function.
- **getTimeRemaining Function** calculates the remaining time from a future deadline
- **startTimer Function** updates the timer display every second based on the remaining time.
- **clearTimer Function** resets the timer to 10 seconds and starts the countdown.
- **getDeadTime Function** calculates the deadline time for the countdown.
- **onClickReset Function** resets the timer to 10 seconds when the “Reset” button is clicked.
- **Return JSX** displays the timer value and a “Reset” button in the UI.

Wrapping Up

React timer is a very useful component in website UI, as it helps in making events more interactive. This tutorial teaches how to create a countdown timer using React JS for your projects. Explained with live working code, this guide provides an easy solution to build a React timer.

We have used React Hooks and `setTimer()` method to create a countdown timer in React JS. Hope this guide, helps you build your first timer using React JS.

“This course was packed with amazing and well-organized content! The project-based approach of this course made it even better to understand concepts faster. Also the instructor in the live classes is really good and knowledgeable.”- **Tejas | Deutsche Bank**

With our revamped [Full Stack Development Program](#): master Node.js and React that enables you to create dynamic web applications.

So get ready for salary hike only with our [Full Stack Development Course](#).

H harsh...

7

Next Article

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

Similar Reads

How to Create Countdown Timer using React Native ?

Our main objective focuses on constructing a straightforward and user-friendly countdown timer that impeccably showcases the remaining time left in terms of years, days, hours,...

4 min read

Event Countdown Timer Using React

Event Countdown Timer is an application where you can create an event with the name, date, and category of the event and it will create a card that will show you the remaining...

5 min read

How To Create A Countdown Timer Using JavaScript

A countdown timer is an accurate timer that can be used for a website or blog to display the countdown to any special event, such as a birthday or anniversary. The basics of a...

5 min read

Create a Event Countdown App with React Native

In this article, we will create an event countdown time application using React Native. The Event Countdown App is a mobile application developed with React Native. It enables...

7 min read

Create Timer Based Phone Silencer App using React-Native

In this project, we will create a mobile application using React Native that acts as a timer-based phone silencer. The app allows users to set a timer, and when the timer finishes, it...

3 min read

[View More Articles](#)

Article Tags :

[React-Questions](#)

[ReactJS](#)

[Web Technologies](#)



GeeksforGeeks

A-143, 9th Floor, Sovereign Corporate
Tower, Sector-136, Noida, Uttar Pradesh -
201305

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)



Company

About Us
Legal
Careers
In Media
Contact Us
Advertise with us
GFG Corporate
Solution
Placement
Training Program

Explore

Hack-A-Thons
GfG Weekly
Contest
DSA in JAVA/C++
Master System
Design
Master CP
GeeksforGeeks
Videos
Geeks Community

Languages

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA
Problems
DSA Roadmap
Top 100 DSA
Interview
Problems
DSA Roadmap by
Sandeep Jain
All Cheat Sheets

Data Science &

ML
Data Science With
Python
Data Science For
Beginner
Machine Learning
Tutorial
ML Maths
Data Visualisation
Tutorial
Pandas Tutorial
NumPy Tutorial
NLP Tutorial
Deep Learning
Tutorial

HTML & CSS

HTML
CSS
Web Templates
CSS Frameworks
Bootstrap
Tailwind CSS
SASS
LESS
Web Design

Python Tutorial

Python
Programming
Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview
Question
Django

Computer Science

Operating Systems
Computer
Network
Database
Management
System
Software
Engineering
Digital Logic
Design
Engineering Maths

DevOps

Git
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

Competitive Programming

Top DS or Algo for
CP
Top 50 Tree
Top 50 Graph
Top 50 Array
Top 50 String
Top 50 DP
Top 15 Websites
for CP

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design
Bootcamp
Interview
Questions

JavaScript

JavaScript
Examples
TypeScript
ReactJS
NextJS
AngularJS
NodeJS
Lodash
Web Browser

Preparation Corner

Company-Wise
Recruitment
Process
Resume Templates
Aptitude
Preparation
Puzzles
Company-Wise
Preparation

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
World GK

Management & Finance

Management
HR Management
Finance
Organisational
Behaviour
Marketing

Free Online Tools

Typing Test
Image Editor
Code Formatters
Code Converters
Currency
Converter
Random Number
Generator
Random Password
Generator

More Tutorials

Software
Development
Software Testing
Product
Management
SEO - Search
Engine
Optimization
Linux
Excel
All Cheatsheets

GeeksforGeeks Videos

DSA
Python
Java
C++
Web Development
Data Science
CS Subjects

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу

[Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

