IT Professional Training Center

Building Single Page Application with Blazor.NET

خلاصه دوره : در سال های اخیر صنعت وب به قدری دچار تغییر و تحول شده است که روزانه شاهد معرفی کتابخانه های متعدد از قبیل Angular, React.js, Vue.js, Aurelia, ... هستیم. تنوع و گستردگی زیاد ابزارها باعث ایجاد نگرانی و سردرگمی در سطح برنامه نویسان وب شده است. یادگیری هرکدام از کتابخانه ها نیز بدلیل داشتن امکانات متفاوت، نیازمند زمان و هزینه خود می باشد. تسلط به زبان JavaScript یا TypeScript کار ساده ای نبوده و در کنار آن آشنا شدن با Nodejs, Webpack, Jasmine, Gulp, ... نیز به عنوان ملزومات توسعه، اجتناب ناپذیر می باشد و البته کار اینجا تمام نمی شود! چراکه لازم است بخش های زیادی از برنامه های Web-Based سمت سرور تولید شوند و لذا باید به فکر انتخاب تکنولوژی های سمت سرور نیز از قبیل... C#, Java, PHP, Payton, نیز باشیم که هر کدام از آنها نیز در جایگاه خود دنیای بسیار متفاوتی دارند! از طرفی اکثر شرکت های نرم افزاری دنبال نیروهای Full Stack می باشند که بتوانند در هر دو طرف (فرانت و سرور) برنامه نویسی نمایند. به جرات می توان گفت که صرفا یادگیری مقدماتی تکنولوژی های مذکور حداقل یک سال زمان نیاز دارد! از طرف دیگر در هر برنامه وب بخش های مشترک زیادی بین فرانت و سرور وجود دارد که لازم است هر کدام از آنها نیز مجددا پیاده سازی شود و خود این موضوع نیز هزینه و زمان تولید برنامه را افزایش میدهد. **پس چه باید کرد؟؟؟** شاید باور کردنی نباشد، ولی شرکت مایکروسافت با ابتکار بسیار جالبی، راهکار جذابی را در اختیار برنامه نویسان خود قرار داده است که نام آن Blazor است!!! تکنولوژی Blazor تمام نگرانی های فوق را برطرف نموده و دریچه جدیدی را پیش روی توسعه دهندگان قرار می دهد. Blazor همه این دوباره کاری ها و نگرانی های توسعه و تولید نرم افزار را از بین می برد و حتی برنامه نویسان سمت سرور نیز براحتی می توانند به دنیای زیبای فرانت وارد شده و UI خود را پیاده سازی نمایند. برنامه نویسی با Blazor در محیط

IT Professional Training Center

یکپارچه Visual Studio و پلتفرم ASP.NET Core 3.0 و Visual Studio Code تجربه مدرن و دلپذیری از تولید نرم افزار را در اختیار شما قرار می دهد.

مخاطبین دوره : تمامی برنامه نویسان که علاقه مند به تولید برنامه های Web-Based می باشند یا کلیه تولید کننده گان نرم افزار که با چالش های تامین نیروی متخصص و هزینه های تولید سمت سرور و فرانت مواجه شدند که می توانند با استفاده از Blazor نرم افزارهای قدرتمندی را تولید نمایند.

پیشنیاز دوره : آشنایی با زبان برنامه نویسی سی شارپ و HTML, CSS

مدت دوره : ۳۰ ساعت

سرفصل دوره :

An introduction to Blazor for ASP.NET Core developers

- An open-source and cross-platform .NET
- Client-side web development
- WebAssembly fulfills a need
- Blazor: full-stack web development with .NET
- Get started with Blazor

Blazor Architecture

Blazor app hosting models

- Blazor WebAssembly apps
- Blazor Server apps
- How to choose the right Blazor hosting model
- Deploy your app

Project structure for Blazor apps

- Project file
- Entry point
- Static files
- Configuration
- Razor components

- Pages
- Layout
- Bootstrap Blazor
- Build output
- Run the app

App startup

Build reusable UI components with Blazor

- An introduction to Razor
- Contents
- Use components
- Component parameters
- Child content
- Attribute splatting
- Event handlers
- Data binding
- Chained bind
- State changes
- Component lifecycle
- OnInitialized
- OnParametersSet
- OnAfterRender
- IDisposable
- Capture component references
- Invoke component method
- @key
- Capture element references
- Templated components
- Child content
- Template parameters
- Code-behind
- Cascading values and parameters
- Manual RenderTreeBuilder logic

Forms and validation

- Input Component
- Data annotation validators
- Filed level validation
- Model level validation
- Validation message
- Validation summary
- Custom validator

Call a WebAPI
- HttpClient and JSON helpers
- Cross-origin resource sharing (CORS)
- HttpRequestMessage

Component libraries

Dependency Injection
- ASP.NET Core dependency injection (DI)
- Add services to an app
- Configuring service lifetimes (Scoped, Singleton, Transient
- Request a service in a component
- Use DI in services
- Utility base component classes to manage a DI scope

Pages and layouts
- Create pages
- Page layout
- Main layout
- Inherits LayoutComponentBase
- Default layout
- Specify a layout in a component
- Centralized layout selection
- Nested layout

Routing
- Router component
- Navigation
- Base URLs
- Route template
- Route parameters
- Route constraints
- Double-asterisk catch-all syntax (**)
- NavLink component
- URI and navigation state helpers

JavaScript interop
- Invoke JavaScript functions from .NET methods
- Call a void JavaScript function
- Detect when a Blazor app is prerendering
- Capture references to elements
- Invoke .NET methods from JavaScript functions

State management

Security and Identity
- Authentication
- Blazor Server authentication

- Blazor WebAssembly authentication
- AuthenticationStateProvider service
- Implement a custom AuthenticationStateProvider
- Authorization
- AuthorizeView component
- Role-based and policy-based authorization
- Customize unauthorized content with the Router component

Handle errors

- Reacts to unhandled exceptions
- Handling exception
- Log errors
- Places where errors may occur

Debug

- Enable remote debugging
- Debug the app

Data access

- Select a database engine (SQL or NoSQL)
- Installing SQL Server or MongoDB
- Create and using Repository Service

Hosting and deplyment