Node-red front document (persian)

فایل های اصلی این پروژه که با آن سر و کار داریم، در قسمت packages/node\_modules میباشد؛ در داخل آن نیز دو پوشه @node-red و node-red قرار گرفته اند؛ @node-red عبارت است از:

1. editor client:

در کل کد های موجود در این پوشه برای مدیریت بخش قابل استفاده برای مشتری یا کاربر میباشد؛ در پوشه locals فایل های مربوط به زبان یا language میباشد؛ یعنی اگر قرار بر این باشد که زبان سایت عوض شود، باید ابتدا در این پوشه json ها ترجمه شوند و دقیقا به همان مدل در پوشه ای قرار گیرند. پوشه src شامل سورس های این قسمت است که فقط عکس و فونت ها را شامل نمیشود، بلکه شامل اسکریپت ها نیز است. بعضی از این اسکریپت ها عبارتند از:

node-red/packages/node\_modules/@node-red/editor-client/src/js/comms.js:

## در ابتدا کل این فایل داخل RED.comms که خود یک فانکشن را صدا میزند میباشد.

## در داخل آن، ابتدا متغیر هایی که قراره استفاده بشه، تعریف شده اند و سپس سه فانکشن کلی وجود دارد که در آخر هر سا تایه آنها return میشوند.

## در فانکشن های subscribe و unsubscribe همانطور که از اسمش پیداست بحث چک کردن میباشد؛ در کل اگر قرار باشد فانکشن subscribe اعمال شود، به منظور ادامه فرایند از قسمتی که مانده است میباشد، و به عکس.

## کار اصلی در فانکشن connectWS اتفاق میوفتد که باز کردن یک سوکت و اعمال تنظیمات آن پورت وظیفه این فانکشن میباشد؛ نام این پورت ws است.

## در حالت کلی وظیفه این فانکشن چک کردن توکن میباشد؛ ابتدا wspath را به عنوان پچ باز کننده سوکت استفاده میکند و سوکت باز شده را در متغیر ws ذخیره میکند.

## این سوکت سه حالت onopen و onmessage و onclose دارد:

## در onopen ارور ها پاک شده و توکن بصورت فایل جیسون در سوکت ارسال میشود، که دارا بودن توکن شرط آن است و دائما در حال چک آن است.

## در onmessage دیتایی در قالب متغیر فانکشنی استخراج میشود که دو مقدار ok و fail آن بررسی میشود؛ اگر که آن مقدار برابر ok باشد، متغیری که مربوط به چک کردن توکن بود با مقدار false مقدار دهی شده و این فاکشن کلی دوباره اجرا میشود؛ در غیر اینصورت یعنی اگر مقدار مورد نظر برابر fail باشد، مقدار جیسونی updateMenu با مقدار true فرستاده میشود. اگر متغیر ورودی فانکشن مقدار دهی نشود، قانون regex جدیدی روی آن اعمال میشود تا خطای مربوط به آن را اصلاح کند.

## در onclose ابتدا چک میشود که مقداری که در فانکشن قبلی به عنوان ok با fail بود و در متغیر جدیدی با مقادیر بولین هست چه وضغیتی دارد؛ اگر ok بود مقدار false ست شده و به عکس؛ این کار برای چک کردن صحت و وجودیت توکن میباشد. اگر توکن موجود نباشد، ابتدا ده بار هر کدام به مدت یک ثانیه و سپس ده بار هر کدام به مدت دو ثانیه و در کل به مدت 30 ثانیه دوباره توکن را چک میکند؛ اگر مشکل توکن حل نشد، به مدت 60 ثانیه اروری را نمایش میدهد که خطای دریافتی را نمایش میدهد؛ مگر اینکه روی دکمه اینجاد شده برای بستن ارور کلیک شود.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/events.js:

## در ابتدا کل این فایل داخل RED.events که خود یک فانکشن را صدا میزند میباشد.

## این فایل شامل سه فانکشن کلی on و of و emit میباشد:

## در فانکشن on متغیر آرایه ای مقدار دهی میشود.

## در فانکشن off متغیر های مقدار دهی شده، پاک میشوند.

## فانکشن emit که RED.events.DEBUG استفاده شده است، بنظر میرسد برای debug بکار میرود و در واقع مدیریت ارور های کنسول برای راهنمایی debug میباشد.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/font-awesome.js:

## در این فایل که RED.nodes.fontAwesome بکار رفته است، مدیریت فونت ها اتفاق میوفتد؛ یعنی فونت ها در این فایل قرار دارند؛ این فایل دارای دو لیست کلی برای متغیرها میباشد که هر کدام در آخر return میشوند. در return دو مقدار به قالب جیسون return میشوند که یکی مقادیر متغیر های موجود در لیست ها ریتورن میکند (valus) و دیگری نام متغیر ها را (keys)؛ فانکشنی که نام متغیر هارا ریتورن میکند، فقط لیست اول را در نظر میگیرد اما فانکشنی که مقادیر را ریتورن میکند، هر دو لیست را در نظر میگیرد.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/history.js:

## این فایل دستورات کلی undo و redu را کنترل میکند؛ شامل دو فانکشن کلی است و ریتورن آن به شکل داده های جیسونی میباشد. در این فایل RED.history استفاده شده است که همه اتفاقات درون آن اتفاق می افتد. همچنین در داخل فانکشن های آن از RED.nodes وRED.group و RED.workspaces و RED.events و RED.editor و RED.view و RED.sidebar.config و RED.subflow و RED.menu استفاده شده است.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/hooks.js:

## این قطعه کد برای مدیریت هوک های بکار رفته، استفاده میشود. در این کد فانکشن های کلی add و remove و clear و has و trigger ریتورن شده است که کارکرد هر کدام از آنها را میتوان به راحتی از روی اسمش تعیین کرد.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/validators.js:

## این فایل تعریف کننده RED.validators میباشد؛ RED.validators یک فایل جیسونی است که سه مقدار number و regex و typeInput را ریتورن میکند؛ در واقع ریتورن هرکدام در داخلشان نوشته شده است. این کد برای چک کردن تایپ فایل موردنظر و سپس تایید آن میباشد.

## در number که خود فانکشنی هست، تایپ ورودی چک میشود که آیا ورودی عدد است یا نه؟ اگر عدد نباشد، ارور مورد نظر را به بک اند میفرستد.

## در regex همان بحث ریجکس چک میشود؛ اگر که ویژگی ریجکس رد شود، ارور را به بک اند میفرستد.

## در typeInput سه مورد چک میشود: 1) ورودی عدد است؟ 2) ورودی جیسون است؟ 3) ورودی مسیج یا msg است؟

## node-red/packages/node\_modules/@node-red/editor-client/src/js/user.js:

## در این فایل، RED.user قرار دارد؛ در انتهای فایل چهار فانکشن به شکل جیسون و به نام های init و login و logout و hasPermission ریتورن میشوند؛ این فایل در کل برای میدیریت لاگین و ساین آپ بکار میرود که در آن حتی توکن نیز چک میشود؛ بطور مثال در logout توکن را revoke یا لغو میکند.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/runtime.js:

## در این کد RED.runtime تعریف میشود؛ در این کد، از RED.menu و RED.settings و RED.events نیز استفاده شده است. این کد زمان اجرا برنامه را کنترل میکند؛ در این کد متغیرهایی برای اجرا و توقف برنامه به کار گرفته شده اند. در ریتورن این کد، init که شامل یک فانکشن برای اعمال کنترل و رفع خطاهای احتمالی است، همراه با یک فانکشن که متغیر شروع زمان را مقداردهی میکند، اعمال میشوند. میتوان گفت کد هایی که با نام runtime در کل پروژه استفاده شده اند، موتور حرکت پروژه هستند.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/red.js:

## این کد تعریف کننده RED میباشد که بیس اصلی همه این فایل هاست؛ در آخر یک فانکشن و یک متغیر را ریتورن میکند که خود آن متغیر نیز تعریفل کننده یک فانکشن دیگر است. فانکشنی که ریتورن میشود همانطور که از اسمش پیداست init برای مقدار دهی اولیه میباشد؛ به نظر میرسد این مقادیر به بک اند فرستاده میشود؛ متغیری که ریتورن میشود نیز دارای یک init میباشد که بیشتر برای چیدن کد های html به کار گر فته میشود. این متغیر که اسمش loader میباشد برای انجام کار های load به کار گرفته میشود.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/polyfills.js:

## این فایل برای بعضی از مرورگرهای قدیمی که پروزه را ساپورت نمیکنند ساخته شده است؛ در واقع وقتی مرورگری قدیمی اروری میدهد، این کد آنرا را حل یا مدیریت میکند.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/plugins.js:

## این فایل RED.plugins را تعریف میکند؛ سه فانکشن دارد و در آخر هر سه را ریتورن میکند:

## registerPlugin که وظیفه آن ست کردن متغیر های موجود در این کد میباشد؛ بصورت ریز به ریز و با توجه به لود ماژول ها کار خود را انجام میدهد.

## getPlugin در واقع آیدی پلاگین را ریتورن میکند.

## getPluginsByType در واقع تایپ پلاگین را تعریف میکند.

## plugin در لغت به معنای افزونه میباشد؛ در این کد از RED.events و RED.\_loadingModule و RED.\_ نیز استفاده شده است.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/nodes.js:

## ...

## node-red/packages/node\_modules/@node-red/editor-client/src/js/main.js:

## این کد فقط document.title را تنظیم میکند؛ در واقع اگر hostname برابر مقادیر صفحه پیشفرض نبود، به document.title اسم hostname غیر پیشفرض را اضافه میکند.

## node-red/packages/node\_modules/@node-red/editor-client/src/js/jquery-addons.js:

## …

## node-red/packages/node\_modules/@node-red/editor-client/src/js/i18n.js:

## این کد تعیین کننده زبان یا همان language سایت است؛ در فانکشنی داده مربوط به زبان را دریافت کرده و با توجه به زبان مورد نظر، کاتالوگ هارا تنظیم میکند؛ در آخر نیز همزمان با تنظیم کاتالوگ ها، زبان مورد نظر را نیز به عنوان متغیر بر میگرداند.

## node-red/test/editor/pageobjects/util/key\_page.js:

## شورت کات های کیبورد در این کد تعریف شده اند. در آخر این سری کدها که در این مسیر قرار گرفته اند، module.exports بکار رفته است که به معنای اکسپورت کردن متغیر یا فانکشن مورد نظر است که در تمام کد قابل دسترسی باشد.

## node-red/test/editor/pageobjects/util/spec\_util\_page.js:

## این کد فقط browser.pause را در قالب فانکشن تعریف کرده است؛ در آخر این سری کدها که در این مسیر قرار گرفته اند، module.exports بکار رفته است که به معنای اکسپورت کردن متغیر یا فانکشن مورد نظر است که در تمام کد قابل دسترسی باشد.

## node-red/test/editor/pageobjects/util/util\_page.js:

## این کد با browser.addCommand سر و کار دارد؛ در کل سه دستور برای مرورگر نوشته شده است که مربوط به کلیک کردن و دریافت تکست و سلکت کردن همرا با زمان انتظار است. browser.pause نیز در اینجا بکار رفته است. این دستور ها در واقع همانند کتابخانه ای برای استفاده هستند که مرورگر از آنها استفاده خواهد کرد. در آخر این سری کدها که در این مسیر قرار گرفته اند، module.exports بکار رفته است که به معنای اکسپورت کردن متغیر یا فانکشن مورد نظر است که در تمام کد قابل دسترسی باشد.

## node-red/test/editor/pageobjects/nodes/node\_page.js:

## در این کد، میتوان گفت که مرورگر چیده میشود؛ مرورگر از کتابخانه تعریف شده مخصوص خود استفاده میکند و ماژول Node را در آخر کد اکسپورت میکند؛ همانند کد های قبلی، در آخر این سری کدها که در این مسیر قرار گرفته اند، module.exports بکار رفته است که به معنای اکسپورت کردن متغیر یا فانکشن مورد نظر است که در تمام کد قابل دسترسی باشد.

## node-red/test/editor/pageobjects/nodes/nodefactory\_page.js:

## این قطعه کد را میتوان به عنوان سازنده یاد کرد؛ چرا که فانکشنی که در اینجا وجود دارد، کار ساخت را با گرفتن تایپ و دادن آیدی انجام میدهد؛ در آخر این سری کدها که در این مسیر قرار گرفته اند، module.exports بکار رفته است که به معنای اکسپورت کردن متغیر یا فانکشن مورد نظر است که در تمام کد قابل دسترسی باشد. در اینجا create اکسپورت شده است.

## ساختن نود جدید:

## برای ساختن نودی جدید، در ابتدا باید سه فایل ایجاد کرد؛ 1).html 2).json 3).js.

## در فایل html در کل سه تگ <script> وجود دارند که هرکدام تایپ و وظیفه مشخص خودش را دارد؛

## اولین اسکریپت با تایپ text/javascript مربوط به تعریف آن نود میباشد؛ در این قسمت در واقع تایپ نودرا ریجستر میکنیم؛ برای این منظور از فانکشن RED.node.registerType استفاده میشود که دو مقدار را قبول میکند؛ اولی برای تعیین تایپ و دیگری فانکشن آن.

## دومین اسکریپت با تایپ text/html و ایونت data-template-name که نام نود را به عنوان مقدار به آن میدهیم، در واقع تعیین کننده قالب html نود میباشد.

## سومین اسکریپت که دقیقا مانند اسکریپت دوم، تایپ و ایونت data-template-name دارد، به اسکریپت help معروف است؛ توضیحات کلی در مورد کارکرد این نود در اینجا نوشته میشود.

## فایل .js شامل همه کارها و فانکشن هاییست که قرار است نود انجام دهد؛ در اینجا نیز نود باید با نام خودش ریجستر شود؛ این کار توسط فانکشن RED.node.registerType انجام میشود. در اینجا اصطلاحات و فانکشن های از قبل تعریف شده نیز کاربرد گسترده ای دارد.

## فایل .json شامل همه داده هاییست که نیاز است استفاده شود؛ از جمله اینها میتوان به نام نود که باید به مدل scope نوشته شود، ورژن node-red و سایر داده ها اشاره کرد. این داده ها قوانین و اصول خاص خودشان را دارا هستند.

## در آخر وقتی که این سه فایل آماده شدند، در ترمینال به مسیر نود رد موجود در سیستم رفته و فایل های جدید را که در پوشه ای قرار دارند را نصب میکنیم، که برای اینکار کافیست مسیر فایل هارا بدانیم و سپس با دستور npm install (file location) نود جدید را نصب کنیم.

## برای دانستن اطلاعات دقیق تر با مثال های کاربردی پیشنهاد میشئود از لینک زیر دیدن نمایید:

## <https://nodered.org/docs/creating-nodes/>