### Aplicação Web

### Arquitectura

A aplicação web representa o estado de jogo partilhado. As interações que a aplicação android despoleta para o servidor são notificadas à Single Page Aplication (SPA), e reflectidas no browser. Para tal foi incluída a biblioteca @aspnet/signalr no projeto de forma a manter a comunicação websocket com o servidor.

A SPA é responsável por apresentar em tempo real as alterações de estado, despoletadas através das interações dos jogadores. É também a SPA que temporiza algumas fases de jogo, ou seja, **contém timers que informam o utilizador do tempo restante para termino da ação corrente e quando o tempo expira notifica o servidor do sucedido de forma a transitar de estado.**

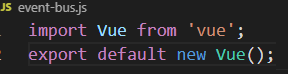
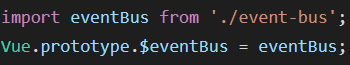
Esta SPA assenta sobre a framework js VueJs. Vue Js tem um sistema que permite renderizar, de uma forma declarativa, os dados na DOM através de uma sintaxe muito simples.



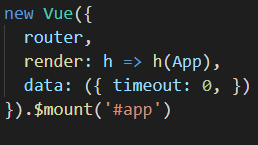
O que acontece aqui é que uma nova instância do Vue é criada, e esta por sua vez, é associada à div com o id “app”. A partir daqui todos os componentes de Vue criados, que representam as páginas da aplicação contendo cada um o seu html, serão injetados dentro desta div. Estes mesmos componentes têm acesso à sua instância, denominada por $root, e poderão aceder a possíveis variáveis globais criadas, neste exemplo “message” é transversal a todos os componentes e acessível através de $root.message.

Tirando partido de esta globalização, e de forma a simplificar o processo de temporização, foi criado um event bus e a variável global timeout.

O event bus é uma outra instância do VueJs, associada à instância principal, que apenas emite o evento “timer-finished” e recebe o evento “clear-timer”. É através destes eventos que os vários componentes da aplicação avançam de estado conforme o tempo de jogo tenha ou não terminado.

Visto que o valor de tempo é enviado nas várias notificações recebidas pelos componentes, foi necessário criar a variável global timeout de forma a ser afetada em diferentes partes da SPA.



De seguida, na Figura 1 e Tabela 1, serão apresentados um diagrama temporal sobre o fluxo da SPA e os vários eventos despoletados tanto pelo servidor como pela Aplicação Web de forma a manter comunicação bidirecional.

Fluxo Aplicacional

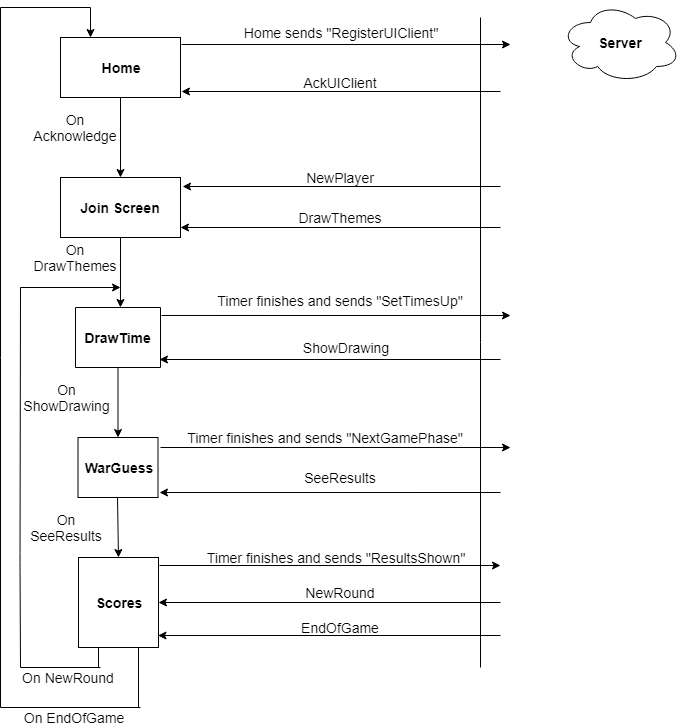


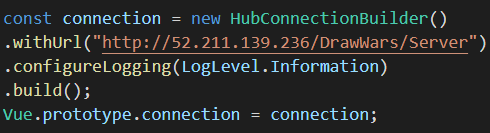
Figura 1

Tabela de eventos SignalR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Origem** | **Destino** | **Nome** | **Ecrã** | **Descrição** |
| Web App | Servidor | RegisterUiClient | Home | Registo do cliente WEB após o click no botão |
| Servidor | Web App | AckUiClient | Home | Confirmação do registo do cliente WEB |
| Servidor | Web App | NewPlayer | Join | Envio do novo jogador |
| Servidor | Web App | DrawThemes | Join | Notificação para avançar de ecrã |
| Web App | Servidor | SetTimesUp | DrawTime | Caso o tempo de desenho termine |
| Servidor | Web App | ReadyToShowDraws | DrawTime | Caso todos terminem de desenhar |
| Web App | Servidor | DrawPhaseLogic | DrawTime | Notifica o servidor que pode ser enviado o desenho |
| Servidor | Web App | ShowDrawing | DrawTime | Avança de ecrã mostrando o desenho |
| Servidor | Web App | PlayerGuess | WarGuess | Envio da guess do jogador |
| Web App | Servidor | NextGamePhase | WarGuess | Caso o tempo de adivinha termine |
| Servidor | WebApp | SeeResults | WarGuess | Avança de ecrã mostrando a pontuação |
| WebApp | Servidor | ResultsShown | Scores | Tempo de visualização de scores termina |
| Servidor | WebApp | NextRound | Scores | Caso ainda haja rondas de jogo |
| WebApp | Servidor | RoundEndedAck | Scores | Tempo de espera para a próxima ronda termina |
| Servidor | WebApp | DrawThemes | Scores | Reinicia o jogo indo para o ecrã DrawTime |
| Servidor | WebApp | EndOfGame | Scores | Não existem mais rondas voltando para Home |

Tabela 1

Após a inicialização interna do VueJs é criada a conexão com o servidor via instanciação de um Hub e o handshake é realizado. A conexão é mantida na instância do Vue de forma a todos os componentes terem acesso e conseguirem comunicar com o servidor.

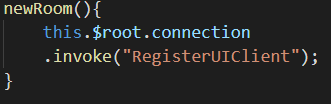
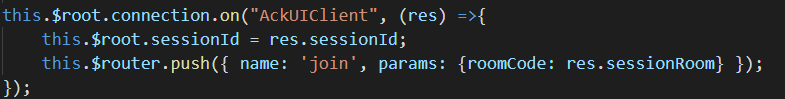


Com o sucesso do handshake é apresentado o ecrã principal, Home.

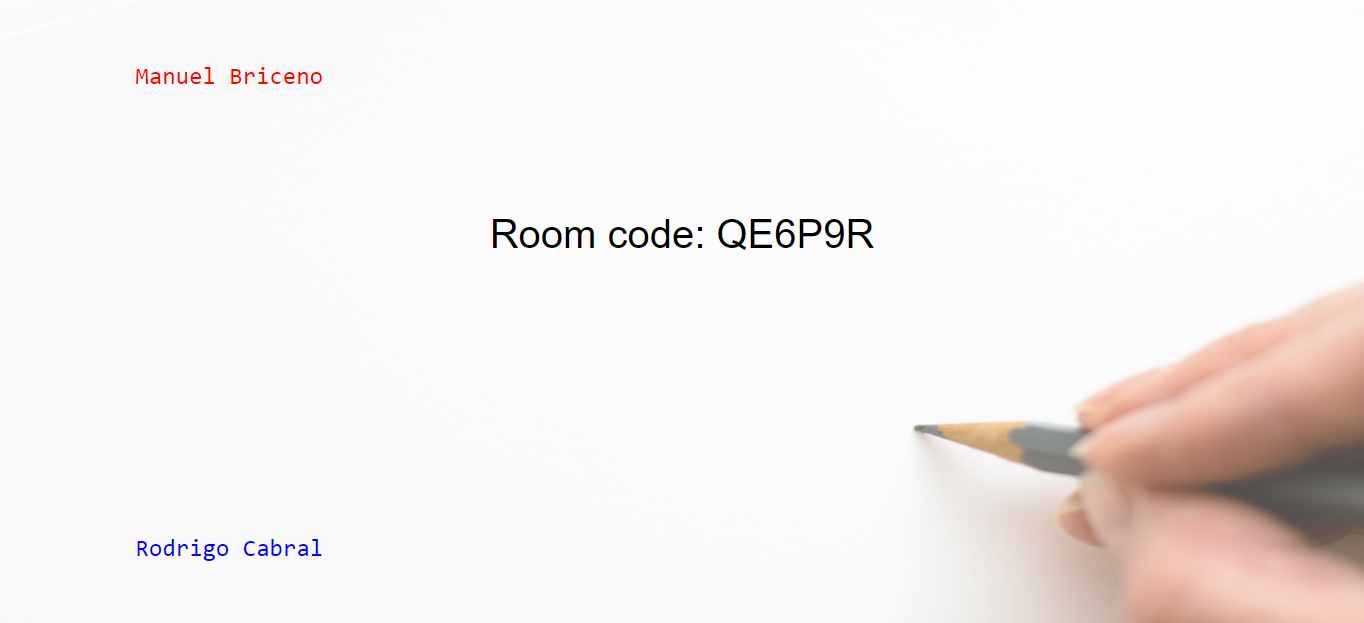


Este ecrã é o ponto de entrada da SPA. No caso de o estado de jogo ser corrompido ou a conexão com o servidor seja quebrada, nomeadamente se o utilizador tentar andar para trás ou refrescar a página, a aplicação é também redirecionada para este ponto. Optamos por esta abordagem visto que podem ser perdidas notificações que tornam o estado aplicacional inconsistente quebrando o fluxo de jogo.

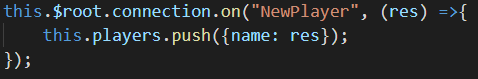
Quando o botão de “New Room” for clicado é lançado o evento “RegisterUIClient” que visa iniciar uma nova sessão de jogo, recebendo do servidor a notificação “AckUiClient” que contém no payload o roomCode. Após receber com sucesso a notificação é feita a navegação para o próximo ecrã, Join, eviando como propriedade o roomCode.

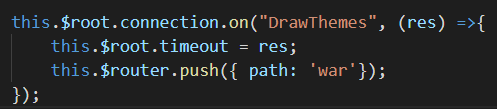
O ecrã de Join apresenta o roomCode e os jogadores.



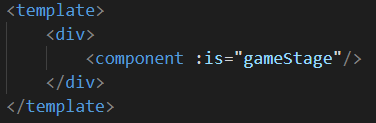
É com este código que os jogadores se ligam à sessão de jogo atual. Quando um jogador insere o código no seu dispositivo android o servidor notifica a SPA com “NewPlayer”, contendo na notificação um novo jogador que irá ser apresentado no ecrã.



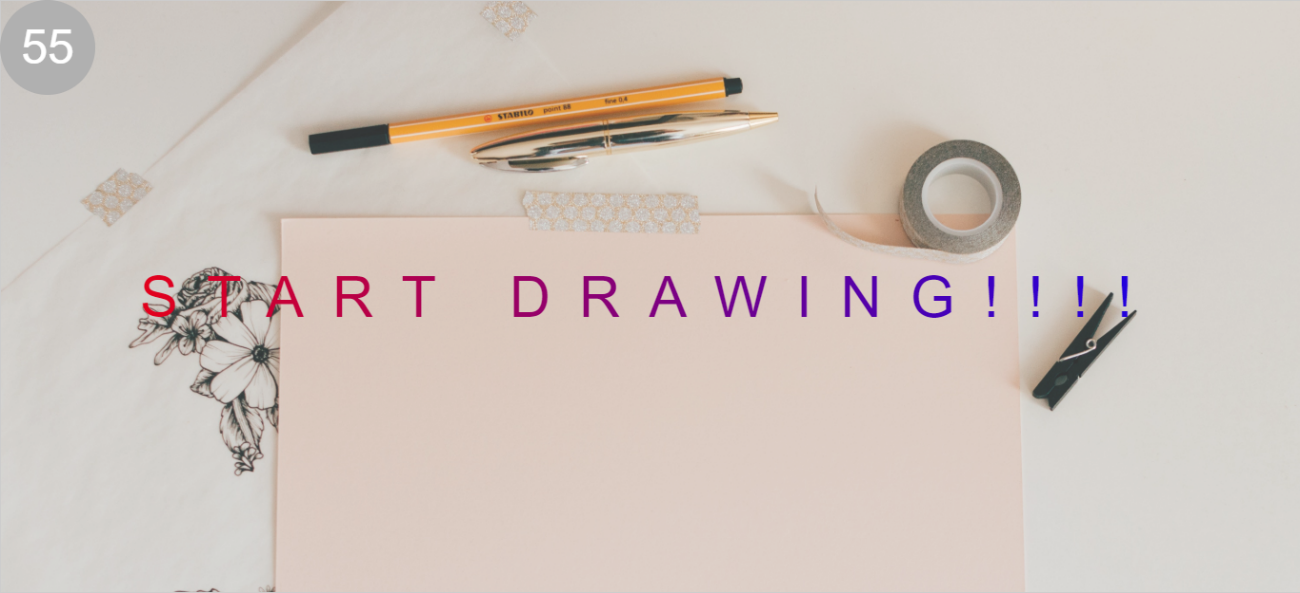
Quando o servidor der início ao jogo a SPA será notificada com “DrawThemes”. Como parâmetro da notificação é passado o valor que irá ser temporizado para a próxima fase de jogo.



A próxima fase de jogo representa um ciclo. O início do ciclo é o início da ronda, ecrã de DrawTime, de seguida é a fase de adivinha, DrawGuess, e por fim a fase de pontuações, Scores. De forma a simplificar o processo de desenvolvimento, e visto que todas estas fases estão interligadas, foi criado um componente base Wars que representa um contentor dos restantes componentes. Neste os componentes são carregados dinamicamente, através da variável gameStage, à medida que o servidor notifica a SPA para a mudança de estado.



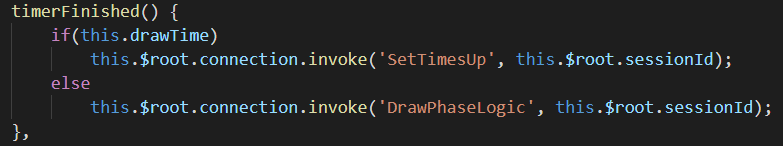
O ecrã de “DrawTime” serve para informar os jogadores que entramos na fase de desenho e podem começar a desenhar nos seus dispositivos android.



Existem 2 formas de esta fase chegar ao fim. A primeira é o tempo estipulado terminar e a SPA lançar o evento “SetTimesUp”. Quando este evento é lançado o servidor recolhe os desenhos em falta, e notifica com “ReadyToShowDraws”. A segunda é todos os desenhos terem sido submetidos a tempo, e o servidor notifica igualmente com “ReadyToShowDraws”. Como parâmetro da notificação, é enviada a quantidade de tempo que o ecrã de transição de fase deve ser apresentado.

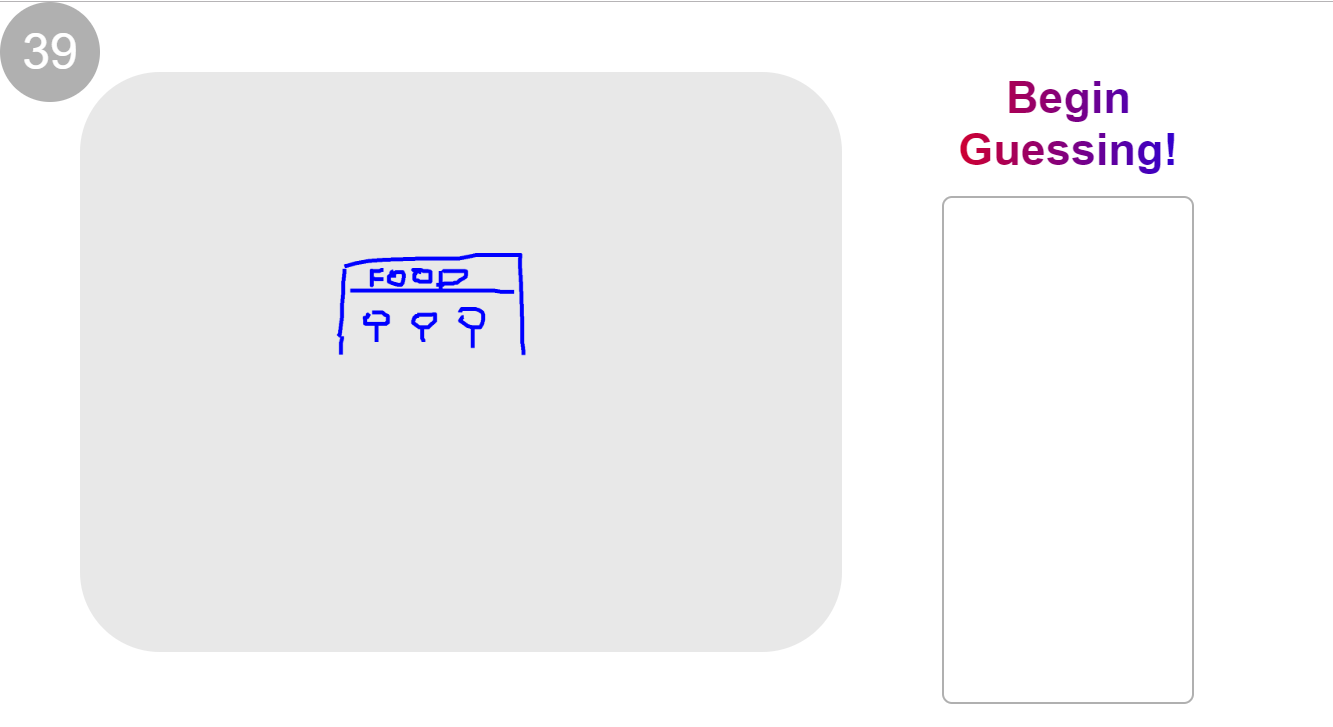


Quando a temporização do ecrã de transição termina, a SPA lança o evento “DrawPhaseLogic” de forma a notificar o servidor que se encontra apta a receber o desenho.



A SPA aguardará passivamente pela notificação “ShowDrawing”, que como parâmetro recebe um url para o desenho a apresentar e um novo valor de tempo para a próxima fase de jogo, e transita para ecrã de WarGuess.

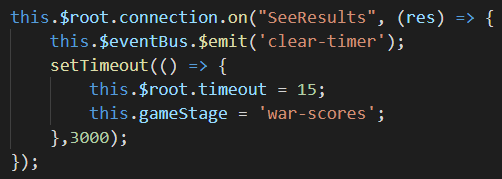
O ecrã de WarGuess é responsável por mostrar o desenho dos jogadores. Contém também uma área onde as adivinhas feitas pelos jogadores são mostradas.



À medida que os palpites vão sendo inseridos no dispositivo android, o servidor vai notificando a SPA com “PlayerGuess”. Esta notificação contém o palpite atual do jogador e se está ou não correto. Caso esteja, em vez do palpite, é apresentado “Correct!” para os restantes não saberem qual a *“guess”* correta.

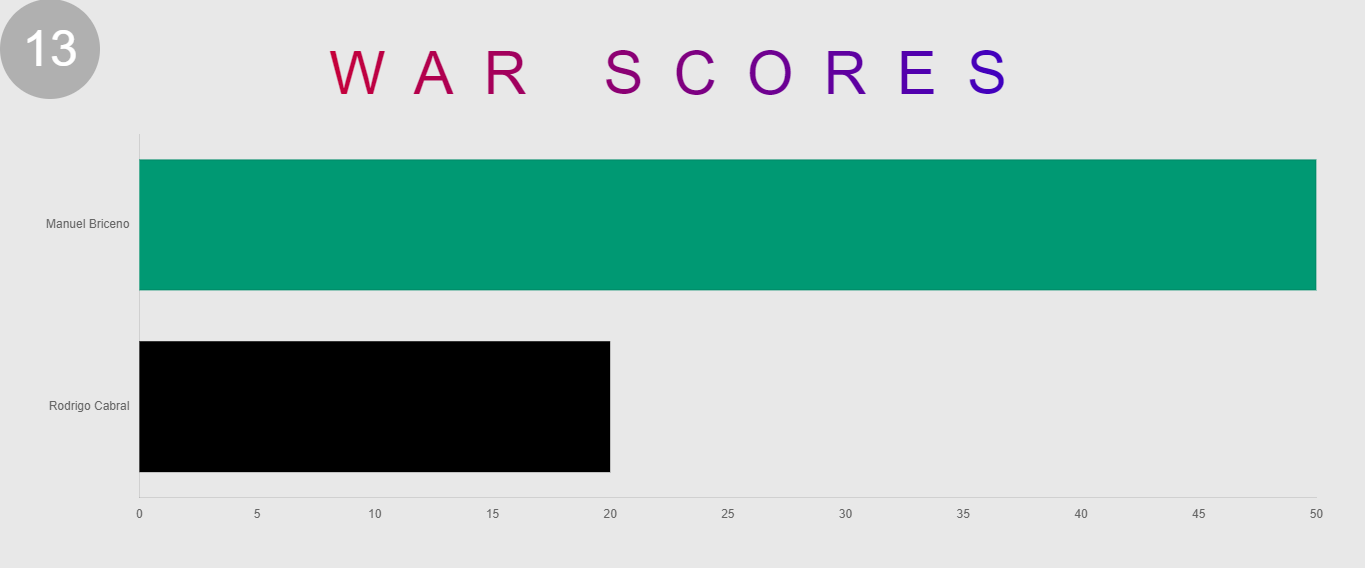
Quando o tempo terminar é enviado “NextGamePhase” pela SPA para informar o servidor do sucedido. Caso o servidor receba “NextGamePhase” ou todos os jogadores acertarem dentro do tempo limite o servidor notifica com “SeeResults” passando como parâmetro as pontuações atuais dos jogadores.

Ao receber “SeeResults” e de forma a que a transição não seja instantânea, ocultando o último palpite submetido, é criado um contador de 3 segundos sendo apresentado no browser um aviso de que não serão aceites mais submissões.

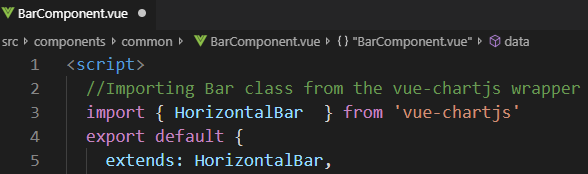


Quando os 3 segundos terminarem, a variável global timeout é afectada com 15(segundos) que será a duração do próximo ecrã “Scores” seguido dessa mesma transição de ecrã.

É no ecrã de “Scores” que as pontuações dos jogadores são mostradas. Para tal utilizamos uma biblioteca externa, *vue-chartjs*, de forma a criar um gráfico de barras.



Esta biblioteca permite de uma forma simplista a criação de vários tipos de gráficos, apenas é necessário importar o tipo de gráfico pretendido, criar um novo componente e extender o gráfico importado passando para a função de render os dados e as configurações do gráfico.

O tempo estipulado para a visualização das pontuações é de 15 segundos. Quando o tempo termina a SPA envia para o servidor “ResultsShown”. O servidor irá responder com 1 de 3 eventos: “ShowDrawing”, “NextRound” e “EndOfGame”.

No caso de os desenhos de todos os jogadores ainda não terem sido apresentados o servidor envia “ShowDrawing” e passa a ser apresentado o ecrã “WarGuess” com o desenho do próximo jogador, seguindo o fluxo explicado anteriormente até ao ecrã “Scores”. Continua assim sucessivamente até todos os desenhos desta ronda terem sido apresentados.

Com o término da apresentação dos desenhos, caso ainda existam rondas de jogo, o servidor envia “NextRound” passando um novo valor de tempo. A SPA ao receber esta notificação apresenta novamente o contador, com o novo valor recebido, e informa os jogadores que uma nova ronda está prestes a começar. Quando o tempo termina é enviado para o servidor o evento “RoundEndedAck”. O servidor ao receber esta notificação envia “DrawThemes” com o devido valor de tempo. Neste caso a SPA reencaminha para o ecrã de “DrawTime” e o ciclo inicial de jogo recomeça.

Quando já não existem mais rondas a serem jogadas, o servidor envia “EndOfGame” em resposta a “ResultsShown” e a SPA reencaminha a aplicação web para o ecrã “Home” onde se pode criar uma nova sessão e iniciar um jogo de novo.