



Documentación del Teclado Visual (KeyboardDisplay)

`KeyboardDisplay.js` es el componente central de visualización del juego. Representa el teclado en pantalla e incluye lógica para mostrar qué teclas se deben presionar, con qué dedo, y cómo retroalimentar visualmente al jugador.

Objetivo principal

Proveer una visualización interactiva y configurable del teclado que:

- Resalte teclas relevantes.
 - Indique qué dedo usarlas.
 - Permita control visual del entorno.
 - Sea reutilizable en distintos modos de juego.
-

Estructura interna

Propiedades principales:

- `this.group`: contiene todos los elementos gráficos (teclas, textos, líneas, manos, botones).
- `this.layout`: define la estructura del teclado (matriz de filas con etiquetas de teclas).

- `this.keyPositions`: mapa con la posición exacta de cada tecla.
- `this.lastKeys`: últimas teclas resaltadas, para redibujar si se modifica el estado.






Opciones configurables (desde constructor o `localStorage`):

- `showExplosionLine` ☒
 - `showFingerDot` ☒
 - `showHands` ☒
 - `fingersOnTop` ☒
 - `soundEnabled` ☒
 - `maxMisses` ☒ (nuevo control visual desde interfaz)
-



Botones integrados

Estos botones están integrados directamente dentro del teclado y permiten activar/desactivar comportamientos visuales y sonoros:

-  Guía (punto y flecha entre dedo y tecla)
-  Manos visibles
-  Sonido
-  Línea de explosión
-  `+` / `-` Control de errores permitidos (`maxMisses`)

Todos estos cambios se guardan en `localStorage`, por lo que persisten entre escenas y sesiones.



Métodos clave

`draw(keys)`

Llama a `drawArray()` o `drawCharacter()` según el tipo de entrada (string o array).

`drawArray(characters)`

- Usa `getKeysForChar()` para determinar las teclas necesarias.
- Filtra para obtener la principal y la pasa a `drawInternal()`.

`drawInternal(keys)`

- Limpia la pantalla.
- Dibuja cada fila de teclas con estilo correspondiente.
- Resalta las teclas necesarias.
- Muestra manos, punto, flecha y línea según configuración.

`addToggleButton()`

Crea los botones visuales interactivos.

`addMaxMissControl()`

Muestra un contador y dos botones (▲ ▼) para ajustar `maxMisses`.



Integración con el juego

Este componente es usado en:

- `TypingFallingScene`
- `TypingTextScene`

- `InterlevelScene`

Cada escena puede invocar `keyboard.draw(letra)` para actualizar lo que debe mostrarse.

Además, otros sistemas pueden leer `keyboard.maxMisses` o leerlo desde `localStorage` para limitar fallos.

Recomendaciones

- Evitar modificar directamente el DOM Phaser fuera de este componente.
- Si se agregan nuevas configuraciones, seguir el patrón: `propiedad`, `botón`, `localStorage`, `redraw()`.
- Para nuevos idiomas o layouts, extender `this.layout` y `getKeysForChar()`.

En desarrollo / ideas futuras

- Posibilidad de personalizar colores por dedo.
- Alternar entre diferentes layouts de teclado.
- Animación de la mano presionando la tecla.
- Visualización de estadísticas por dedo o zona del teclado.

Actualizado según versión: `KeyboardDisplay.js` con integración de controles persistentes, botones visuales y selector de errores (`maxMisses`).