Marp a primera vista

Nos empeñamos en realizar tutoriales pero en este caso vamos a realizar una simple revisión a un software novísimo: **Marp.** Esta entrada viene a colación por un "tuit" del sr. Aitor León desde Sevilla, España donde hace un autorecordatorio sobre un software del cual nosotros no habíamos escuchado antes y que nos llamó la atención porque le aplicó la etiqueta #MarkDown:

https://twitter.com/2Ait8r/status/873121378207121408

Nosotros con mucho respeto le preguntamos qué era ese programa y amablemente nos respondió:

https://twitter.com/2Ait8r/status/873163773644861440

https://twitter.com/2Ait8r/status/873189783207632896

Ni tardo ni perezosos indagamos y he aquí lo que pudimos aprender, lo cual humildemente compartimos con el mundo entero de habla castellana.

Inicio y desarrollo de Marp.

Un paso para comenzar.

Ya bien dice el refrán que todo viaje, por muy largo que sea, comienza con un solo paso: el 11 de febrero de 2016 (hace una gran cantidad de tiempo en términos de ordenador japonés) se escribió el comienzo de Marp, el cual originalmente se llamaba **MDSlide** en su versión 0.0.1.

[cc lang="javascript" tab_size="2" lines="40"]
{
"name": "mdslide",
- "version": "0.0.0",
+ "version": "0.0.1",
"description": "Presentation writer with markdown",
"main": "main.js",
"scripts": {
[/cc]

Desarrollo: nace Marp.

En mayo de 2016, luego de duro trabajo, MDSlide se transforma en **Marp** y la numeración de versiones correlativo corresponde a V0.0.6; al cambiar de nombre nace un repositorio nuevo y de MDSlide no quedan rastros... *dice uno* **porque bien sabemos que el protocolo Git es**

distribuido y por ahí en algún disco fijo debe haber una copia, je, je, je. El logotipo es el ahora conocido y la verdad ya estaba bastante maduro: soporte para emoji, ecuaciones matemáticas (eso nos impresiona) e imágenes (recordad que se basa en MarkDown: se trabaja con teclado).

Ruta de trabajo.

Para octubre de 2016 se retoma el desarrollo debido a que escuchando las sugerencias de los usuarios se replantea todo y liberan la versión 0.0.9 y se declara un plan de trabajo para liberar la tan ansiada versión 1.0 . También debemos tomar en cuenta que **Electron**, el entorno de programación utilizado, también ha sido actualizado y eso "empuja" a **Marp** en cierta medida y a pesar de suficientes "pull requests" (copias del código fuente modificado por terceros que lo devuelven mejorado y/o con sugencias) que recibe el sr. Hattori, él ratifica que sigue en las riendas del proyecto y espera mejorarlo mucho gracias a sus usuarios y colaboradores. *Pero en este punto reconoce que el modo de presentación es la parte más difícil* ***y es algo que aún no han logrado pulir, pues es esa la característica que estuvimos buscando nosotros y no vemos por ningún lado (más adelante mostraremos su uso con ejemplos en castellano).*** La "discusión" es viva y candente (en inglés) e incluso se le llega a comparar con software privativo que lleva décadas en el mercado.

Última versión a la fecha de escribir este artículo.

En noviembre de 2016 liberan la versión 0.0.1 que es la versión que descargamos y probamos pero al abrirlo nos encontramos con una versión compilada a lenguaje de máquina y con una extraña versión 1.3.8 **¿En qué momento nos perdimos... de algo?** De todos modos no especulemos nada aún, en la parte del licenciamiento veremos bien si realmente es "software libre" o de "fuente abierta".

Autoría.

Así nació Marp de las manos del Ingeniero de Software Web Yuki Hattori, graduado en « ?????????? » ("Public Hakodate Future University") y quien actualmente trabaja en la empresa "Speee, Inc." en Tokio, Japón.

El Ingeniero Hattori maneja una gran cantidad de lenguajes de programación (vamos que desayuna **PHP**, merienda con **CSS**, almuerza con **Ruby**, la merienda de la tarde la pasa con **Node** y cena con **Rails**) y nosotros que pensábamos que algo programamos, pues nos quedamos cortos (44% en otros lenguajes de programación ¿cuáles serán? Acá hace algo con SVG dinámico y en este enlace lo podéis probar y modificar cortesía de Phil Maurer).

Licencia de uso.

El repositorio oficial de **Marp** lo podéis encontrar en este enlace, y de sus múltiples secciones hemos leído y analizado la información para el presente artículo, esparamos os sea útil en algún modo.

En junio de 2016 el proyecto tiene un mes con su nombre actual **Marp** y deciden colocar correctamente la licencia de uso que corresponde a la del Instituto Tecnológico de Massachusetts (**MIT**) la cual no puede ser un simple enlace web, se debe colocar el encabezado con el año y el nombre del autor para luego copiar fielmente la licencia de uso del MIT y luego, si se quiere, el enlace web correspondiente.

De hecho múltiples organismos conservan una copia de dicha licencia y la más socorrida es la de "The Open Source Initiative"; vale la pena recordar entonces que una cosa es **software libre** y otra cosa es **código fuente abierto**: para no ser puristas el software libre es más antiguo y solo tiene 4 reglas básicas (aunque ha evolucionado) y el código fuente abierto deriva su filosofía del software libre y tiene 10 reglas básicas.

En julio de 2016 se reconoce que el proyecto contiene un trabajo realizado con anterioridad y que también posee licencia de uso del MIT: Katex el cual está escrito en JavaScript y CSS para representar gráficamente las tediosas fórmulas matemáticas. *Nosotros que estudiamos ingeniería mecánica y nos tocó estudiar matemáticas avanzadas damos fe del tremendo trabajo de presentar informes y no tener un software adecuado para ello*. Da un poco de vergüenza decirlo pero era rapídisimo realizarlo de forma manuscrita y presentable en hojas blancas que lo horroroso que quedaban con las impresoras de matríz de punto y el software que disponíamos. **Hemos progresado** hasta el punto que hoy tenemos presentaciones en un lenguaje de marcación ligero y sencillo, que puede representar las matemáticas como son *¡y de paso se puede exportar a formato pdf, increíble!*

En este mismo mes se decide retirar el anuncio de licencia de uso de *underscore.js* ya que no se sigue utilizando como componente.

Página web oficial de Marp.

En este enlace se tiene una presentación bien cuidada acerca del software y orientada a los usuarios normales, no programadores. Podemos ver su características, un enlace al repositorio, un enlace para descargarlo, una rápida introducción a su uso, una guía a ejemplos más avanzados y una declaratoria de licencia de uso. **Lo que nos llama poderosamente nuestra atención es la inserción de dos objetos web que presentan por diapositivas a Marp, dichos objetos fueron hechos en** *Deckset* -software hecho en Alemania-**el cual está también basado en MarkDown...** ¿casualidad? Bueno acá el software de fuente abierta supera al privativo, pues no exporta en formato pdf... y podremos agregarle funcionalidades que consideremos haciendole un "fork" y haciendo nuestro propio proyecto (conservando la licencia de uso del MIT, por supuesto).

"Instalando" Marp.

Pues que si llamamos "instalar" a descargar un archivo y descomprimirlo, vale, "cuela" como instalación. Esta entrada no es tan compleja como otras que hemos hecho que analizamos lo más a fondo que podemos su código fuente, pero no os preocupéis, no perderemos de vista a Marp en el futuro. Repetimos, el repositorio está en esta dirección.

Usando Marp.

Pilares fundamentales de Marp.

- Escribimos en MarkDown las presentaciones (y también podemos ver *el resultado del MarkDown*).
- Se pueden escoger *temas* y fondos de pantalla para las presentaciones.
- Soporta caracteres UNICODE (mejor conocidos como *emojis*).
- Se puden escribir ecuaciones matemáticas con hermoso estilo.
- Se puede exportar a formato PDF.

Haciendo presentaciones en MarkDown.

Retrospectiva.

Admitámoslo: realizar presentaciones es tedioso para los que no tenemos el talento para los gráficos. Somos empedernidos usuarios de la ventana terminal y por eso nos agrada mucho el MarkDown y su simple sintaxis para obetener una apariencia básica del HTML... ¡Pero esperen, hay más!

Desde que Github popularizara el Git con su alojamiento gratuito para proyectos públicos (que de acuerdo a la licencia de uso que tengan tendrán diferentes clasificaciones) y agregaron el uso de MarkDown acompañando a los repositorios con su respectiva sección Wiki, que también utiliza cierto "sabor" de MarkDown, ***y ahora con una ligera variante podremos hacer presentaciones rápidamente sobre nuestro trabajo ya hecho pues... ¡BINGO! ??***

Hala, manos a la obra.

Aunque entre las carpetas descomprimidas hay una llamada "locales" hay un archivo llamado **es.pak** cuyo contenido abrimos con **gedit** y está escrito en castellano, *no hallamos opción para cambiar el idioma inglés.* Revisando el código fuente no hallamos referencia a dichos archivos .pak (hay bastantes idiomas) lo que imaginamos es que son "paquetes" o plantillas que se utilizan masivamente para internacionalizar nuestras aplicaciones... o simplemente será nuestra imaginación.

Dicho todo esto, os escribiremos en inglés las referencias que hagamos al programa Marp.

No nos afanéis buscando teclas de acceso directo -para hacer cualquier cosa- excepto las siguientes:

Marp, como es de esperarse, utiliza la misma extensión de fichero de los documentos **MarkDown**: .md (tal vez presentemos algún vahído porque estamos bien acostumbrados a que las presentaciones tienen sus propias extensiones según el software que las manipule).

Lo primero es abrir un archivo nuevo, vamos al menú desplegable "**File**" -> "**New file**" (o pulsamos las teclas CTRL+N). Sin más preámbulo se abre un "lienzo" de trabajo presto a escribir en el. En este lienzo a la izquierda escribimos nuestro código y a la derecha veremos inmediatamente nuestro trabajo de tres maneras distintas:

- En modo de presentación, diapositiva por diapositiva (modo por defecto).
- En modo de presentación, diapositivas en hilo o "carrousel".
- *En modo MarkDown, tal como se vería al publicarla en la página web de GitHub, por ejemplo **(¡interesante!).***

Títulos y subtítulos.

Inician con un símbolo numeral "#" seguido de un espacio y el texto para título principal y agregando dos o más numerales juntos -sin espacios- los subtítulos correspondientes (hasta un máximo de seis subniveles):

*Nota 1: **Marp** como buen intérprete del **MarkDown** "original" soporta el uso de "=" y "-" (uno o más) en la línea inmediatamente inferior del texto para mostrar un encabezado de primer y segundo nivel. Pero si necesitamos de un tercer nivel necesitaremos tres numerales "###"... así que no vale la pena este método os aseguro.*

*Nota 2: **Marp** aplica a las imágenes un "realzamiento" su utilizamos numeral (es) delante de la declaración de imágenes, **sin embargo "=" y "-" no funcionan en estos casos*** (no le busquen lógica a esto, por favor, no sabemos si es un "bug" o es algo hecho a propósito).

Realizando una segunda diapositiva.

Como véis el código MarkDown funciona y no es obejtivo de esta entrada mostraros su sintaxis *por lo tanto vamos a enfocarnos en los comandos propios de Marp, que son apartes del MarkDown con el propósito de presentar nuestras diapositivas.*

Tal como insertamos una línea horizontal en Markdown (tres guiones seguidos al principio de una línea) así agregaremos una segunda diapositiva, cuidando de dejar una línea en blanco antes y después:

Enumerando las diapositivas.

Para agregar el número de diapositiva en la esquina inferior derecha solo debemos agregar el siguiente código.

A partir de la diapositiva donde insertemos la orden serán numeradas hasta el final, si queremos que todas están numeradas la posicionaremos en la primera diapositiva; *por el contrario, si queremos que una diapositiva en particular NO sea numerada modificaremos la directiva anterior agregandole un asterisco para indicar que se le aplique solo a esa diapositiva.* Por supuesto, usaremos el valor verdadero o falso a nuestro gusto o necesidad.

Directivas globales y directivas locales.

Lo que acabáis de ver "encerradas" entre comentarios tipo HTML "" por supuesto no se visualizan en nigún navegador o intérprete de MarkDown sin embargo **Marp** obtiene sus instrucciones de allí en el formato de una instrucción por línea separadas por dos puntos ":". Dichas instrucciones son llamadas *directivas* y como vimos serán globales -para todas las diapositivas, el documento completo- o solo para la diapositiva donde se encuentre la directiva que establezcamos *siempre y cuando sean antecedidas por un asterisco*.

Dado el caso que nos equivoquemos en la sintaxis pues simplemente **Marp** no las toma en cuenta ni ejecuta y no arroja ningún tipo de mensaje de excepción.

Directiva "$theme" y "template".

Marp soporta plantillas predeterminadas para nosotros dedicarnos a lo valioso de escribir los datos en las diapositivas, lo demás son "adornos de repostería". Al momento de escribir esto estaba disponible el tema **gaia** y de manera adicional con la directiva **template:invert** se tiene una inversión en los colores. Esto queda guardado en el documento de diapositivas para que **Marp** automáticamente la muestre al volver abrir el documento. *Nota: si por el menú desplegable escogemos "View"->"Theme"->"default" -o cualquier otro tema que haya disponible- **Marp** obviará la directiva que hayamos establecida dentro de la diapositiva y mostrará la que le hayamos ordenado.* Verbigracia:

Marp directiva "footer".

Con esta directiva podremos establecer una frase en pie de página ya sea en todas las diapositivas o solo en la que le coloquemos la directiva, recordad el asterisco hace a la directiva local, mirad:

Directiva $size

Esta directiva puede ser confusa porque se refiere a **tamaño** sin embargo presenta también unos valores que indican **proporción** pero si buscamos la lógica ambas son excluyentes mutuas y explicamos por qué: todo depende si nuestra presentación será por pantalla (primera opción para el 99% de los casos, vamos que hablamos de presentaciones por diapositivas). El otro asunto es si necesitamos imprimir dichas presentaciones.

- Para la proporción tenemos "4:3" predeterminado para monitores normales y la mayoría de proyectores traen este valor por defecto y "16:9" para monitores "multimedia" -llamados así porque las películas son grabadas en esa proporción panorámica-.
- Para el tamaño de papel tenemos desde A0 hasta A8 y B0 hasta B8, os adelantamos que el "tamaño carta" viene a ser A4 (mirad abajo en las fuentes consultadas el resto de los medidas de papel).
- Por los dos puntos anteriores es que decimos que son mutuamente excluyentes.
- *Esta directiva solamente es global, recomendamos declararla al principio para un orden efectivo* (hicimos pruebas y el asterisco lo que hace es inutlizarla).
- Es indiferente que usemos mayúsculas o minúsculas en los tamaños de papel.
- Marp lee de izquierda a derecha y el primer tamaño válido de papel o proporción lo aplica, el resto de los caracteres lo desecha (debería NO aplicar el formato o al menos indicar la excepción, pensamos).
- ***No conseguimos hacer funcionar el prefijo documentado "-portrait", recordad que aún es un software en desarrollo y sus gazapos tendrá.***

Directivas $width y $height.

De necesitar un tamaño personalizado de papel, estas dos directivas son las adecuadas al caso y ambas heredan todo de la directiva **$size** por ello no os ponemos gif animado aunque si os podemos decir que si que funciona (¡vamos, animaros a descargar y probar **Marp**!) y las opciones disponibles son:

- **px**: píxeles por defecto, no es necesario colocarle "px".
- *Por la razón anterior las siguientes unidades debéis colocarla juntitas sin espacio(o)s o si*

no lo interpretará como píxeles.
- **cm**: centímetros.
- **mm**: milímetros.
- **in**: pulgadas "inches".
- **pt**: puntos.
- **pc**: picas.

En realidad dichas unidades son las utilizadas por **CSS** *excepto* ***em*** *(tamaño de fuente) y "%";* abajo en "Fuentes consultadas" (en idioma portugués) tenéis un enlace si queréis aprender más por vuestra cuenta.

Directiva "$prerender".

Antes de entrar de lleno con las imágenes os presentamos esta directiva especialmente hecha para indicarle a **Marp** que haga un preproceso de las imágenes de fondo en el caso de que éstas sean muy grandes, *aligera el proceso con una carga previa de datos.*

Comando para mostrar imágenes.

Así como lo leeís, ya finalizamos las directivas y volvemos a los comandos de **MarkDown** en este caso os indicamos lo siguiente:

- Para comenzar a insertar una imagen comenza por una línea en blanco donde colocareís **** tal cual, incluyendo los paréntesis, son 8 caracteres en total.
- Dentro de los corchetes rectos (paréntesis rectos) colocaréis el texto alterno si no se puede cargar la imagen, podéis dejarlo vacío pero el uso de los corchetes es obligatorio, de lo contrario no muestra imagen alguna.
- Dentro de los paréntesis (curvos) pero fuera de las comillas dobles el nombre del fichero de la imagen (si reside en la misma carpeta de donde ejecutásteis **Marp** o una URL válida, local o internet).
- La razón de utilizar una imagen desde internet es para mostrar los últimos valores o cifras relacionados con el tema de la presentación, de lo contrario llevad tus imágenes junto con el documento **Marp**
- Entre la URL de la imagen y las comillas dobles debéis dejar un espacio -el mismo que consideramos al principio en el conjunto de 8 caracteres iniciales de declaración.
- Dentro de las comillas dobles -que sí son opcionales- podemos colocar una breve descripción acerca de la imagen o una frase nemotécnica que podramos visualizar al pasar el puntero por encima de la imágen, tras lo cual al cabo de menos de un segundo se colocará una pequeña ventanita en blanco sobre negro con el texto que coloquemos.

Comando para mostrar imágenes de fondo.

Muchas personas querrán, en vez de utilizar o crear tema alguno, simplemente fijar una o más imágenes como fondo en las presentaciones. He aquí que **Marp** se toma la ligera licencia de tomar la sintaxis de las imágenes -en **MarkDown**- con la excepción que en los corchetes rectos colocamos el comando **[bg]** y por supuesto la URL local o remota de la imagen en sí. Dentro de los corchetes rectos acompañando a "bg" más un espacio podemos colocar un procentaje para redimensionar la imagen e incluso podemos colcoar varias imágenes siempre y cuando los porcentajes sumen 100% para garantizar visibilidad completa de cada imagen. El comando "bg" no soporta mayúsculas, disfrutad nuestras pruebas:

Comandos para mostrar caracteres *emoji.*

Si se sorprenden por haber colocado los *emoji* como subsección de las imágenes preparaos a recibir una historia bien larga... vamos a tratar de resumirla lo mejor posible.

Por allá en los años ochenta cuando nosotros comenzamos a manipular computadoras y luego entramos a estudiar en la Universidad de Carabobo... *¡EA, ¿PARA DÓNDE VAÍS, QUEDAROS Y ESCUCHAD!* ?

...? os contábamos que apenas teníamos computadoras de 16 bits XT y luego vinieron las de 32 bits -286, 386, etc- y en los años 90 se populariza -y hay hardware para manejarlo- el UNICODE que no es más que la representación codificada y gráfica -dibujada con papel y lápiz- de unos 65 mil y pico de caracteres- de la mayor parte de los alfabetos ideados por la humanidad. Allí se tuvo previsión de crear unos símbolos que representaban caras con diferentes estados de ánimo y objetos comunes, *el gran problema era dibujarlos por pantalla* ***Y ESO AÚN HOY EN DÍA ES DIFÍCIL DE HACERLO ¿por qué decimos esto?***

Está bien, los sistemas operativos modernos manejan los códigos extendidos de UNICODE, con tan "poderosos" aparatos que tenemos faltaba más, faltaba menos. ***Pero a nivel de hardware, de tarjeta madre, aún no hemos visto que tengan dicho UNICODE grabado en firmware, los asiáticos han hecho hermosos BIOS gráficos pero que en realidad lo que hacen es cargar en memoria un mini sistema operativo con interfaz gráfica para su idiomas basados en ideogramas: AÚN NO SUCEDE QUE EL CPU ENVIE A UN MONITOR CUALQUIERA EL CÓDIGO BINARIO DE UN CARACTER UNICODE (EXTENDIDO) Y EL MONITOR TENGA "MEMORIA" PARA SABER COMO SE DIBUJA ¿Nos seguís el hilo?*** {Nota: todo monitor e impresora de matriz de puntos tiene en memoria cómo dibujar los 255 caracteres ASCII originales, pero hasta allí llegan, e incluso las impresoras a inyección de tinta y láser ¡perdieron esa capacidad!}.

Más aún, la genialidad de *Steve Jobs*, el fundador de *Apple Computer* -hoy la empresa más poderosa del mundo, incluso por encima de Microsoft e IBM juntas- **fue el poner en práctica el trabajo desarrollado por Xerox Palo Alto (empresa aliada mutua con Apple) de fuentes tipográficas: pequeñísimas imágenes que computadoras con la suficiente potencia pueden**

dibujar rápidamente en pantalla imágenes que nosotros identificamos como "letras"... ¿y todo este discurso para qué? ?

Todo este discurso es para deciros que aún los sistemas operativos modernos NO POSEEN soporte nativo y por ende los navegadores web tienen que obtener de algún lado esas pequeñísimas imágenes que nosotros lamamos "letras" *o, para este caso,* ***emoticons.***

Repetimos, lo simplificamos lo más que pudimos, en serio, **no somos licenciados en computación** pero como nos tocó vivir esa experiencia en carne propia, la evolución de los ordenadores (y escribimos, que algo queda para la posteridad de las generacioens futuras) *pero a ese nivel da este tema.*

Volviendo al tema de los ***emoticons*** o ***emoticones*** para castellanizar el término, revisamos el código fuente de **Marp** y observamos que utiliza dentro de sus dependencias (a esto nosotros lo llamamos *librerías dinámicas*) algo llamado "markdown-it-emoji" y a su vez buscamos en el código fuente de ese software unos *"comandos"* que permitan especificar los *emoji* en **Marp**.

Para **Marp** un emoticon o emoji viene encerrado entre par de dos puntos ": :" y dentro, sin espacios viene la palabra clave con la cual se busca y se dibuja (imaginamos que al compilar **Marp** todas esas imágenes vienen a nuestro disco duro en el fichero ejecutable *luego investigamos y nos dimos cuenta que* a este proceso ELECTRON lo llama "Application Distribution ", con el archivo **app.asar** del cual podrán saber y entender su forma de almacenar la información en su propio repositorio).?

Ya para finalizar el tema de **Marp** y los emoticones, esos comandos o palabras claves (que no trae documentados cuántos ni cuales son) que os dijimos, los copiamos, encerramos entre ": :" y los pegamos en **Marp** para producir la siguiente captura *(otra nota: los emoticones se ven afectados en tamaño si están precedidos como título "#", notad eso)*:

```
:angry::blush::broken_heart:  :confused::cry::frowning:  :heart::imp::inn
ocent:    ---    :joy::kissing::laughing:  :neutral_face::open_mouth::rag
e:    ---    :smile::smiley::smiling_imp:  :sob::stuck_out_tongue::sungla
sses:    ---    :sweat::sweat_smile::unamused:  # :wink:
```

Podéis obtener una lista ampliada, nemotécnica, en este enlace cuyo encabezado anuncia las páginas web y/o aplicaciones que utilizan esos *emoji*. Por ejemplo **:cat:** es más fácil de memorizar que U1F408, caracter unicode, y que funciona sin problemas con **Marp.** En dicha página hacéis click en el emoji que os guste y es copaido la palabra clave con su par de dos puntos, y pegáis en **MArp.** Un código que no veréis es el de GitHub :**octocat:** (combinación de pulpo con gato) así

que no es una lista definitiva, por ahora.

Nota "final" sobre el tema?: estos emoticones que véis son los de Wordpress, que Twitter tiene los suyos propios (acá está cómo enlazar el CDN a vuestras páginas web y/o aplicaciones web y acá un ejemplo de vista previa de las imágenes obtenidas **que por curiosa coincidencia son los mismos del mapa de caracteres de Ubuntu ? je, je, je**)

Actualizado el domingo 11 de junio de 2017.

> Una lista completa de los códigos nemotécnicos de Marp los encontraréis en esta dirección: https://github.com/markdown-it/markdown-it-emoji/blob/6a65f5183fe01452198050409883 41bf76ade32e/lib/data/full.json
>
> Ofrecemos disculpas por no haber investigado lo suficiente. Esa sí es una lista completa de los caracteres *emoji* y soportados por **Marp** porque forma parte de sus componentes y compilación.

Enlaces web.

Los enlaces web o hipervínculos en **Marp** los podemos declarar de forma implícita y explícita (forma recomendada):

- De manera implícita: si escribís "http://" más un caracter cualquiera, **Marp** lo convierte de inmediato en un enlace a la derecha en la ventana de visualización. También cuela "www." más al menos dos caracteres.
- De forma explícita: como dicta el **MarkDown**: un par de corchetes rectos (opcionales) donde colocaremos el texto del enlace y un par de paréntesis (curvos) donde colocaremos el enlace web en sí mismo.

Antes de hablar de las ecuaciones matemáticas...

...debemos indicaros que si necesitamos mostrar *el código fuente en MarkDown a nuestra*

audiencia debemos encerrarlas entre triple comillas simples y **Marp** los interpretará como texto simple, es decir, no lo "ejecutará", no hará "parser".

Ecuaciones matemáticas.

Y acá lo que nos apasiona de tanto en tanto: las matemáticas, especialment el cálculo infinitesimal. Para ello **Marp** tiene un suplemento que debemos ingresar entre par de símbolos de pesos (o lo que sería con el paso del tiempo como símbolo de dólar). Cualquier texto que escribamos lo representa tal cual pero si lo antecedemos de cierta sintaxis comenzaremos a escribir nuestras fórmulas (no os preocupéis, no nos extenderemos mucho a favor de vuestra paciencia).

Fórmula de Euler.

- Claro, cualquier texto lo representa tal cual...
- ... pero si lo antecedemos con un "_" tendremos un subíndice.
- Con "^" lo convertiremos en superíndice.
- Para agrupar estos subíndices o superíndices los encerramos entre llaves "{}"
- Para multiplicar términos: "\cdot".
- A "\cdot" le podemos agregar paréntesis para agrupar más términos.
- Para mostrar tres puntos suspensivos: "\cdots".
- ¡También podemos combinar "\cdots()"!
- Ya con esto podremos representar la famosa *fórmula de Euler:*

Nota: cada fórmula matemática debe ocupar una o más líneas completas, no podemos intercalar fórmulas en una línea, por ejemplo. Opinamos que debería hacerlo y hasta podemos clonar el repositorio y empezar a programar, para eso es el software libre.

Identidad de Euler.

Símbolos y signos.

Es una lista larga, larguísima como la matemática misma pero los más comunes símbolos matemáticos tienen sus propios códigos:

- Para representar ? usamos "\pi".
- Para representar ?: "\phi"-
- Para ?: "\theta"
- Infinito: "\infty".

Operadores matemáticos simples y avanzados:

- División: "\div".
- Fracción: "\frac{}" -colocamos el numerador entre llaves y el denominador a continuación.
- Sumatoria: "\sum" -la podemos combinar con superíndices y subíndices.
- Raíz cuadrada: "\sqrt" más el número -o llaves, o paréntesis.
- Paréntesis grandes, ideales para encerrar fracciones: "\Bigl(" y "\Bigl)".
- Para representar integrales simples: "\int", dobles "\iint", triples "\iiint", integral de superficie "\oint".

Fórmula de cálculo de la constante *e.*

Exportar a formato PDF.

Probamos la exportación de archivos a formato PDF y no tuvimos ningún tipo de problema, si acaso un detalle que "no recuerda" los últimos archivos abiertos o guardados, "archivos recientes" les decimos. Incluso los emoticones fueron exportados y hermosamente dibujados -claro si originalmente fueron hechos en SVG nunca pierde calidad alguna al "pasar" de un lado a otro.

Conclusiones.

El software se ve realmente prometedor y aunque tiene la opción F11 pra verlo a pantalla completa *no es lo mismo que un programa de presentación que se precie como tal: debe ser capaz no solo de usar las flechas de dirección del teclado para avanzar diapositivas sino que le falta aún programador de tiempo, música* ***y lo más importante, ser capaz de detectar -y escoger- los múltiples monitores y/o proyectores que tengamos conectados a nuestro ordenador.***

Esperamos os haya sido útil este vuestro -y nuestro- trabajo para la ampliación de conocimientos y saberes. ;-)

Fuentes consultadas:

En idioma castellano:

- "Editores de Markdown: MarkdownPad 2 y Marp (II)" por Aitor León.

En idioma inglés:

- «Web Engineer Yuki Hattori» at LinkedIn.
- «Yuki Hattori» at Qiita.

- «SVG Loading icons» by Phil Maurer (aurer.co.uk) at codepen.io
- «Marp creates PDF presentations from Markdown» by Mike Williams.
- «Marp: Markdown Presentation Writer» at github.io
- «Basic writing and formatting syntax» at GitHub.
- «Dimensions Of A Series Paper Sizes»
- «Dimensions Of B Series Paper Sizes»
- «Markdown: Basics» by John Gruber.
- «Using Katex with examples».
- «Function support in Katex».

En idioma portugués:

- «Folhas de Estilo Web Dicas & truques CSS».

En idioma japonés (nos disculpáis cualquier "fe de errata" al traducir):

- «??????» (comienzo del blog).