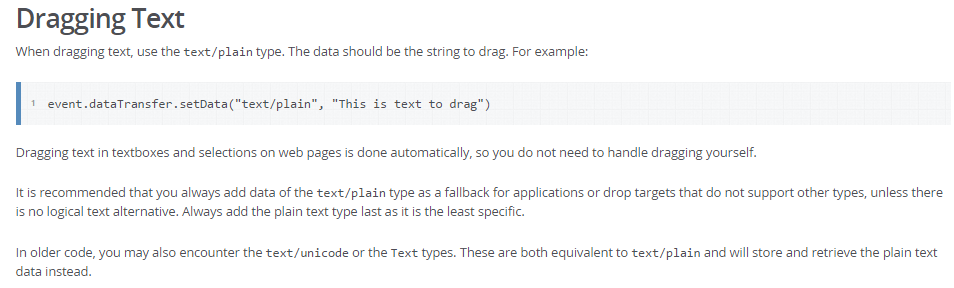
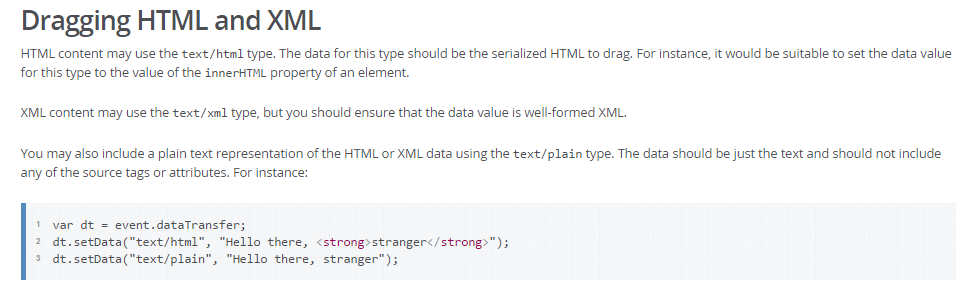
* **Recommended drag types:**





* **Drag Events:**
* dragstart
* dragenter
* dragover
* dragleave
* drag
* drop
* dragend
* **Pasos:**
* The draggable atribute:
* Draggable attribute to **true**, en aquel elemento que quieras sea draggable.
* Añadir un detector para el evento **dragstart**.
* Establecer los datos de arrastre del detector definido anteriormente.

Ej:



* En el caso de que el atributo se omita o se dé como false, el elemento dejará de ser draggable y en su lugar, todo el texto será seleccionado.
* Starting a Drag Operation:

Cuando un usuario empieza a arrastrar, el evento dragstart se dispara. Dentro del dragstart event, se pueden especificar los datos de arrastre, efectos, o la imagen que quieres que se muestre. Aunque sólo los primeros son necesarios, ya que muchas veces las imágenes y los efectos por defecto son adecuados.

* Drag Data:

Todos los drag events tienen una propiedad llamada *dataTransfer* que se usa para contener los datos de arrastre. Estos a su vez contienen dos tipos de información: el tipo o formato, y el valor. El formato es a type string (**text/plain**, **text/uri-list, image/jpeg**) y el valor es la palabra. Durante el arrastre, para los events **dragenter** y **dragover**, se debería utilizar los types of data para comprobar si el drop se permite. Puedes crear tb tu propio tipo, los más comunes se encuentran en el artículo **Recommended Drag Types**.

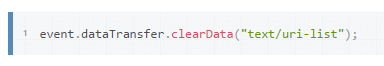
Para establecer los datos dentro de la dataTransfer, se usa el método *setData*. Se requieren dos argumentos: *type of data* y *data value*:



En este caso el valor es “test to drag” y su formato “text/plain”. Se pueden tb proporcionar datos en múltiples formatos, para hacer esto, se debe llamar al método *setData* múltiples veces con diferentes formatos. Desde el más hasta el menos específico. Ej:



Puede borrar los datos utilizando el método *clearData*, que toma el type of data para eliminarlo.



* Setting the drag feedback image:

Puedes usar el *setDragImage()*, para especificar una imagen personalizada:



* Drag effects: *copy, move, link*

Puedes especificar una de estas tres operaciones para un origen de arrastre, estableciendo la propiedad *effectAllowed* dentro de un detector de eventos *dragstar*.

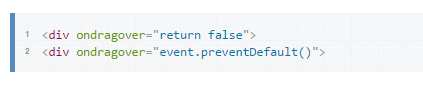


En este ejemplo, sólo *copy* está permitido; pero puedes combinar valores en diferentes vías:

* **none** operation is permitted.
* **copy**  only
* **move** only.
* **link** only.
* **copyMove**
* **copyLink**
* **linkMove**
* **all** copy, move or link.
* Specifying Drop Targets:

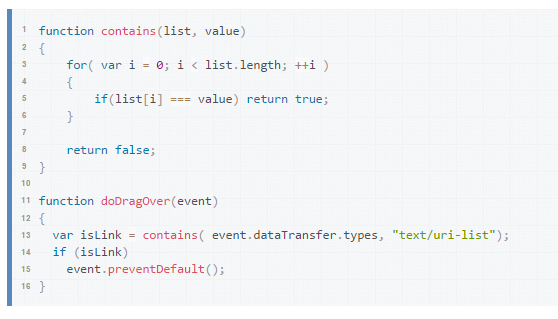
Un oyente para los *dragenter* y *dragover* events se usan para indicar un drop válido abatible, es decir, lugares donde los elementos arrastrados se puedan dejar.

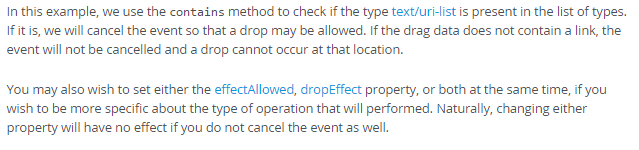
Si deseas permitir el drop, se debe prevenir la manipulación por defecto mediante la cancelación del evento. Esto se puede hacer devolviendo **false** desde un atributo definido, o llamando al método *event.preventDefault* del evento. Este último sería el más factible.



Al llamar al método *event.preventDefault* durante los eventos *dragenter* y *dragover* se indica el lugar del drop del elemento sobre esa localización. Sin embargo, se debe llamar a este método sólo en determinadas situaciones, en las que se cumpla la condición. Por lo que si no se cumple, no se cancela el evento, y un drop no ocurrirá allí donde el usuario suelte el botón.

Es más común para aceptar o rechazar un drop en función del tipo de datos del drag data en la transferencia de datos. Los types, son una lista de string types que hemos añadido al comenzar el drag, de mayor a menos.





* Drop Feedback:

Hay varias maneras para indicar a un usuario que un drop está permitido para cierta localización.

* Performing a Drop:

Cuando el usuario suelta el ratón, el drag and drop termina. Si el ratón deja caer el elemento en una posición válida, el drop se hará satisfactoriamente; mientras que si la operación es cancelada, no se producirá ningún evento drop.

El método *getData* se utiliza para recuperar los elementos de nuevo.



El método *getData* toma un argumento, type of data (“text/plain”), el cual recuperará el valor del string que se fijó cuando fue llamado el *setData* al principio de la operación Drag. En el caso de que no existiera el tipo de data, se devolverá un string vacío.

* Finishing a drag:

Una vez que el drag se ha completado, un evento *dragend* se dispara en la fuente del drag