

Drag and drop a text selection

Drag and drop a text selection from this paragraph. Drag and drop any part of this text to the drop zone. Notice in the code: there is no need for a `dragstart` handler in case of text selection: the text is added to the clipboard when dragged with a key/name equal to "text/plain". Just write a drop handler that will do an `event.dataTransfer.getData("text/plain")` and you are done!

This paragraph is not selectable, however. Look at the CSS in the source code.

Drop some text selection here.

er in case of text selection:

Drag and drop a text selection

There is no need to add a `dragstart` handler on an element that contains text. Any selected text is automatically added to the clipboard with a name/key equal to "text/plain". Just add a `dropevent` handler on the drop zone and get the data from the clipboard using "text/plain" as the access key:

```
function drop(target, event) {  
    event.preventDefault();  
    target.innerHTML += event.dataTransfer.getData('text/plain');  
};
```

EXAMPLE: SELECT SOME TEXT AND DRAG AND DROP THE SELECTION IN THE DROP ZONE

Try it in your browser below (select text, then drag and drop it into the drop zone): or [play with it at CodePen](#):

Drag and drop a text selection from this paragraph. Drag and drop any part of this text to the drop zone. Notice in the code: there is no need for a dragstart handler in case of text selection: the text is added to the clipboard when dragged with a key/name equal to "text/plain". Just write a drop handler that will do an event.dataTransfer.getData("text/plain") and you are done!

This paragraph is not selectable, however. Look at the CSS in the source code.

Drop some text selection here.

Complete source code from the example:

```
<html lang="en">
<head>
  <style>
    .box {
      border: silver solid;
      width: 256px;
      height: 128px;
      margin: 10px;
      padding: 5px;
10.     float: left;
    }
    .notDraggable {
      user-select: none;
    }
  </style>
  <script>
    function drop(target, event) {
20.     event.preventDefault();

    target.innerHTML +=event.dataTransfer.getData('text/plain');
    };
  </script>
```

```

</head>
<body>
  <p id="text">
    <b>Drag and drop a text selection from this paragraph</b>.
    Drag and drop any
    part of this text to
    the drop zone. Notice in the code: there is no need for a
    dragstart handler in case of
    text selection:
    the text is added to the clipboard when dragged with a
    key/name equal to "text/plain".
    Just write a
    drop handler that will do an
    event.dataTransfer.getData("text/plain") and you are
    done!
  </p>
36. <p class="notDraggable">
37.   This paragraph is not selectable however. Look at the CSS in
   the source code.
38. </p>
    <div class="box" ondragover="return false"ondrop="drop(this, event)">
      <p>Drop some text selection here.</p>
    </div>
  </body>
</html>

```

Here, we use a CSS trick to make the second paragraph non-selectable, by setting the `user-selected` property to none.

In the next chapter of Week 3, we will see how to drag and drop files!