C1.1 : Recenser et identifier les ressources numériques

Tout au long de l'apprentissage, nous avons besoin de documentation (même après !), surtout quand on apprend un langage informatique. Pour nous aider dans cette tâche, il existe tout un tas de ressources numériques.

Premièrement, les documentations liées directement au langage ou au Framework utilisé. Dans mon cas, pendant mon stage de formation, j'ai dû passer pas mal de temps le nez dedans :

Utiliser le Hook d'état

Les *Hooks* sont une nouveauté de React 16.8. Ils permettent de bénéficier d'un état local et d'autres fonctionnalités de React sans avoir à écrire de classes.

La page d'introduction présentait les Hooks avec cet exemple :

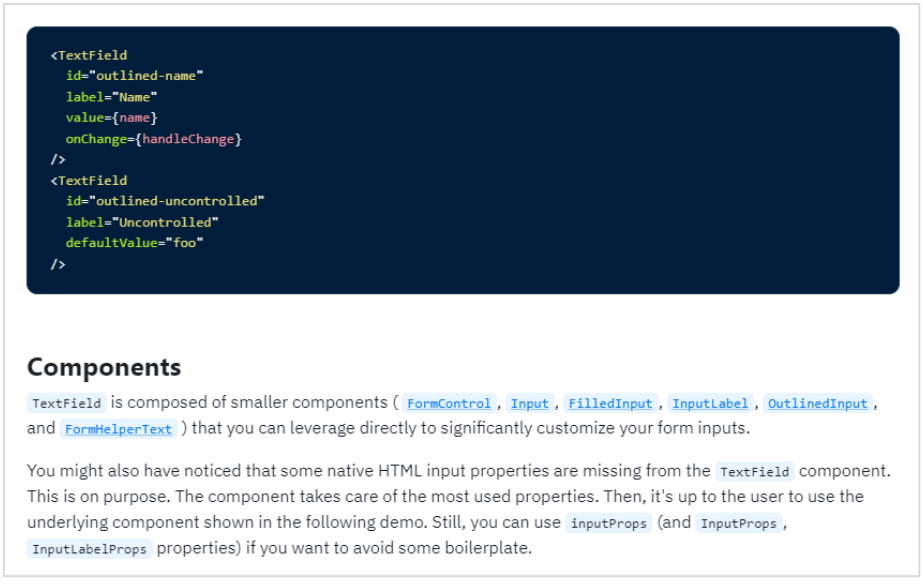
```
import React, { useState } from 'react';

function Example() {
  // Déclare une nouvelle variable d'état, que l'on va appeler « count »
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>Vous avez cliqué {count} fois</p>
      <button onClick={() => setCount(count + 1)}>
        Cliquez ici
      </button>
    </div>
  );
}
```

Pour commencer à comprendre comment fonctionnent les Hooks, comparons ce code avec un exemple équivalent à base de classe.

Utilisation des hooks « use State » du langage React.



```
<TextField
  id="outlined-name"
  label="Name"
  value={name}
  onChange={handleChange}
/>
<TextField
  id="outlined-uncontrolled"
  label="Uncontrolled"
  defaultValue="foo"
/>
```

Components

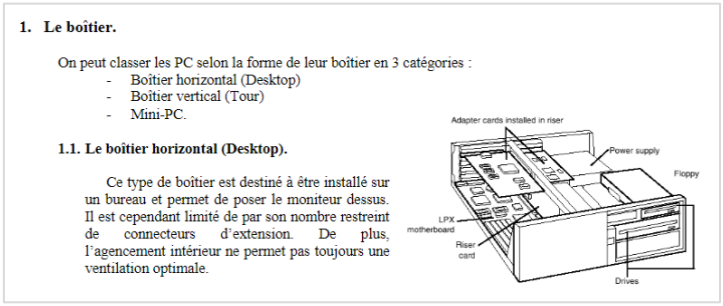
`TextField` is composed of smaller components (`FormControl` , `Input` , `FilledInput` , `InputLabel` , `OutlinedInput` , and `FormHelperText`) that you can leverage directly to significantly customize your form inputs.

You might also have noticed that some native HTML input properties are missing from the `TextField` component. This is on purpose. The component takes care of the most used properties. Then, it's up to the user to use the underlying component shown in the following demo. Still, you can use `inputProps` (and `InputProps` , `InputLabelProps` properties) if you want to avoid some boilerplate.

Utilisation des textfields contrôlés du framework « Material UI »

Nous avons aussi besoin d'aides plus « généralistes », afin de comprendre comment fonctionne, au mieux, notre environnement de travail. En début d'année scolaire, nous avons pu avoir des cours sur les composants d'un ordinateur/serveur. Nous avons donc pu voir en détail de quoi était composé une « *UCT* ».

Nous avons donc commencé par découvrir « l'enveloppe » que sont les boîtiers et leurs alimentations :



1. Le boitier.

On peut classer les PC selon la forme de leur boîtier en 3 catégories :

- Boîtier horizontal (Desktop)
- Boîtier vertical (Tour)
- Mini-PC.

1.1. Le boîtier horizontal (Desktop).

Ce type de boîtier est destiné à être installé sur un bureau et permet de poser le moniteur dessus. Il est cependant limité de par son nombre restreint de connecteurs d'extension. De plus, l'agencement intérieur ne permet pas toujours une ventilation optimale.

Cours sur le boîtier d'une UCT

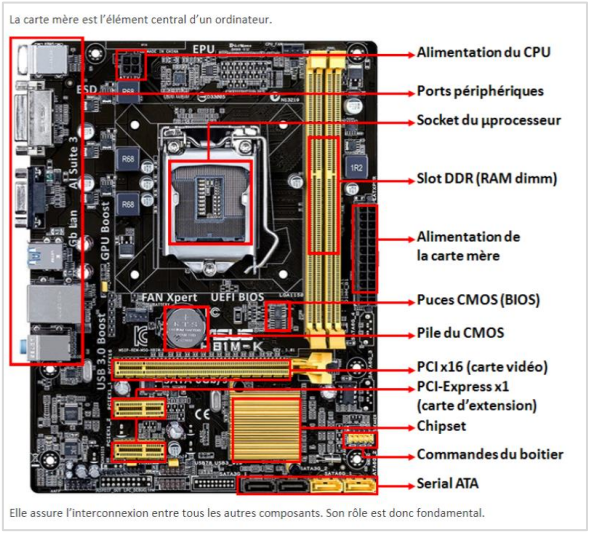
2.1. Rôles de l'alimentation.

Le rôle de l'alimentation est essentiellement de convertir le courant électrique alternatif du secteur EDF 230V alternatif en différentes tensions continues utilisables par l'ordinateur :

3,3V 5V 12V -12V -5,5V

Cours sur l'alimentation d'une UCT

Puis, nous avons vu le rôle d'une carte mère :

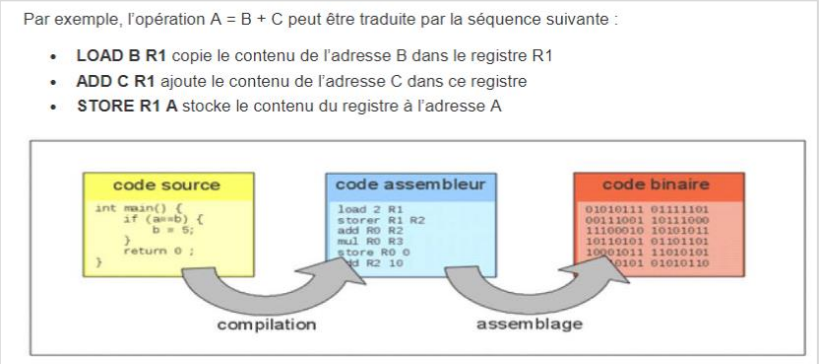


La carte mère est l'élément central d'un ordinateur.

Elle assure l'interconnexion entre tous les autres composants. Son rôle est donc fondamental.

Cours sur la carte mère d'une UCT

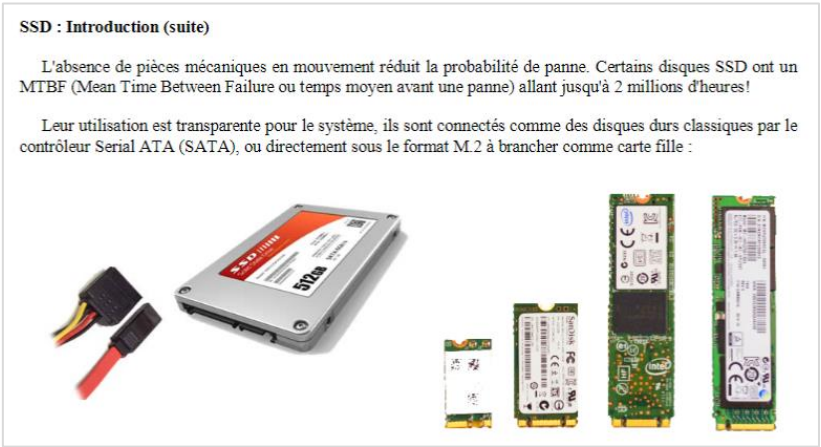
Ensuite, nous avons étudié le microprocesseur :



Cours sur le CPU d'une UCT

Pour enfin terminer sur la mémoire (*morte et vive*), indispensable dans une configuration :

Cours sur la mémoire morte d'une UCT



Cours sur la mémoire vive d'une UCT