DWA_07.4 Knowledge Check_DWA7

---

1. Which were the three best abstractions, and why?

```
const bookPrieview = ({ author, id, image, title }) => {
  const element = document.createElement("button");
  const isElement = element instanceof HTMLElement;
  if(!isElement) throw new Error('[dataAtrr not set]');

  element.className = "preview";
  element.setAttribute("data-preview", id);

  element.innerHTML = /* html */
  `
        <img
            class="preview__image"
            src="${image}"
        />

        <div class="preview__info">
            <h3 class="preview__title">${title}</h3>
            <div class="preview__author">${authors[author]}</div>
        </div>
    `;

  return element;
};

books.slice(0, BOOKS_PER_PAGE).forEach(book => {
  starting.appendChild(bookPrieview(book));
});
```

In the above example the class that I created is bookPrieview. It follows the notion of SOLID Principles.

S - the above class follows a notion of performing one execution preventing the code from being altered at the root class.

This is a great function which assist with code complexities. As when other people work on the same code if they by accident try and alter the class to a different value it won't function, it also prevents merge conflicts.

O - the abstraction created provides for the code to be extended into other functions without the modification to the class taking place

---

2. Which were the three worst abstractions, and why?

---

3. How can The three worst abstractions be improved via SOLID principles.

By adhering to the principles and also redesigning your flawed abstractions as they are easier to fill with bugs.

By not altering the root class and adding to its functionality could be a huge improvement to the created abstractions.

Understanding the intent of the created Abstractions and whether or not modifications to it has an impact on other data stored and if more addition to the class will cause a ripple effect to alter data.
