Sure, let me explain these concepts in depth:

**1. `innerText` vs `innerHTML`:**

- \*\*`innerText`\*\*: It represents the text content of an HTML element. It strips out any HTML and returns only the visible text. For example:

```html

<div id="example">Hello <strong>World</strong></div>

```

Using `innerText` on the above div:

```javascript

const text = document.getElementById("example").innerText;

console.log(text); // Output: "Hello World"

```

- \*\*`innerHTML`\*\*: It represents the HTML content inside an element. It returns the full HTML content, including any tags. For example:

```html

<div id="example">Hello <strong>World</strong></div>

```

Using `innerHTML` on the above div:

```javascript

const htmlContent = document.getElementById("example").innerHTML;

console.log(htmlContent); // Output: "Hello <strong>World</strong>"

```

### 2. Explanation of Code Snippets:

#### a. `const pinInput = (document.getElementById("pinInput") as HTMLInputElement).value;`

- `document.getElementById("pinInput")`: This gets the DOM element with the ID `"pinInput"`.

- `(document.getElementById("pinInput") as HTMLInputElement)`: This part is using type casting (`as HTMLInputElement`) to inform TypeScript that the element retrieved is an input element.

- `.value`: This retrieves the current value of the input element. `pinInput` now holds the value of the input element with the ID `"pinInput"`.

#### b. `document.getElementById("pinForm")!.style.display = "none";`

- `document.getElementById("pinForm")`: This gets the DOM element with the ID `"pinForm"`.

- `.style.display = "none";`: This sets the CSS `display` property of the element to `"none"`, effectively hiding the element from the page.

#### c. `document.getElementById("options")!.style.display = "block";`

- `document.getElementById("options")`: This gets the DOM element with the ID `"options"`.

- `.style.display = "block";`: This sets the CSS `display` property of the element to `"block"`, making the element visible on the page.

#### d. `const amount = parseInt((document.getElementById("withdrawAmount") as HTMLInputElement).value);`

- `document.getElementById("withdrawAmount")`: This gets the DOM element with the ID `"withdrawAmount"`.

- `(document.getElementById("withdrawAmount") as HTMLInputElement)`: Similar to the first example, it typecasts the retrieved element as an input element.

- `.value`: This retrieves the value entered in the input field.

- `parseInt(...)`: This converts the retrieved value (which is a string) to an integer.

#### e. `const forms = document.querySelectorAll('form'); forms.forEach(form => form.reset());`

- `document.querySelectorAll('form')`: This selects all `<form>` elements on the page and returns a NodeList.

- `forms.forEach(form => form.reset());`: This loops through all the `<form>` elements in the NodeList and resets their values, effectively clearing the input fields inside those forms.