JavaScript Code Explanation and Commands

1.

```
document.getElementById("send_it").addEventListener("click", function(e){
    e.preventDefault();
});
```

- document.getElementById(): Selects an HTML element by its ID.
- .addEventListener("click", callback): Adds a function to run when the element is clicked.
- function(e): An anonymous function that receives the event object e.
- e.preventDefault(): Prevents the default action of the event, here it stops form submission/reload.

2.

```
let currentuser = null;
let balances = [];
let currentbalance = null;
```

- Declares three variables using let:
  - currentuser: stores the username of the logged-in user.
  - balances: an array to hold balance data.
  - currentbalance: holds the currently logged-in user's balance.

3.

```
function login() {
    fetch('login.json')
        .then(response => response.json())
        .then(data => {
            const loginArray = [];
            data.forEach(entry => loginArray.push(entry));

            const inputUser = document.getElementById("username").value;
            const inputPassword = document.getElementById("password").value;

            const isValid = loginArray.some(entry =>
            entry.user === inputUser && entry.password === inputPassword
        );

        if (isValid) {
            currentuser = inputUser;
            document.getElementById("welcome").innerHTML = "Login successful. Welcome, " +
inputUser;
            document.getElementById("working_area").innerHTML = "What would you like to do today
" + inputUser + "?";
            document.getElementById("login_area").style.display = "none";
            console.log('Login successful!');
        } else {
            document.getElementById("welcome").innerHTML = "LOGIN UNSUCCESSFUL! Please
try again!";
            console.log('Invalid credentials.');
        }

        document.getElementById("username").value = "";
        document.getElementById("password").value = "";
    })
    .catch(error => {
        console.error('Error loading JSON:', error);
    });
}
```

Key commands:

- function login() { ... }: Declares a function named login.
- fetch('login.json'): Asynchronously requests the login.json file.
- .then(response => response.json()): Converts fetch response to a JS object/array.
- Array.forEach(): Iterates over an array.
- document.getElementById().value: Gets input values.
- Array.some(): Returns true if any array element meets a condition.
- innerHTML: Sets HTML content.
- style.display = "none": Hides an element.

- console.log(): Logs to console.
- .catch(): Catches errors.

4.

```
function balance() {
    if (!currentuser) {
        document.getElementById("balance").innerHTML = "Balance";
        return;
    }

    fetch('balances.json')
        .then(response => response.json())
        .then(data => {
            balances = data;
            const userBalance = balances.find(entry => entry.user === currentuser);

            if (userBalance) {
                currentbalance = Number(userBalance.balance);
                        document.getElementById("working_area").innerHTML = "Your balance is: EUR" +
userBalance.balance;
                console.log("Balance:", userBalance.balance);
            } else {
                document.getElementById("balance").innerHTML = "Balance not found for user.";
            }
        })
        .catch(error => {
            console.error('Error loading balances:', error);
        });
}
```

- if (!currentuser): Checks if no user logged in.
- fetch() and .then(): Loads balances.json asynchronously.
- Array.find(): Finds first matching element.
- Number(): Converts to number.

- Updates UI with balance or error.

5.

```
function deposit() {
    if (!currentuser) {
        document.getElementById("working_area").innerHTML = "Please log in first!";
        return;
    }

    if (currentbalance === null) {
        document.getElementById("working_area").innerHTML = "Balance not loaded. Please check your balance first.";
        return;
    }

    const new_deposit = Number(prompt("How much would you like to deposit?"));

    if (isNaN(new_deposit) || new_deposit <= 0) {
        alert("Invalid deposit amount!");
        return;
    }

    currentbalance += new_deposit;
        document.getElementById("working_area").innerHTML = "Deposit successful! Your updated balance is EUR" + currentbalance;

    const userIndex = balances.findIndex(entry => entry.user === currentuser);
    if (userIndex !== -1) {
        balances[userIndex].balance = currentbalance;
    } else {
        alert("User not found in balances.");
    }
}
```

- prompt(): Gets user input via popup.
- isNaN(): Checks if value is not a number.
- alert(): Shows alert popup.
- +=: Adds amount to balance.
- Array.findIndex(): Finds index of user in balances.
- Updates balance in memory and UI.

6.

```
function withdraw() {
    if (!currentuser) {
        document.getElementById("working_area").innerHTML = "Please log in first!";
        return;
    }

    if (currentbalance === null) {
        document.getElementById("working_area").innerHTML = "Balance not loaded. Please check your balance first.";
        return;
    }

    const withdrawalAmount = Number(prompt("How much would you like to withdraw?"));

    if (isNaN(withdrawalAmount) || withdrawalAmount <= 0) {
        alert("Invalid withdrawal amount!");
        return;
    }

    if (withdrawalAmount > currentbalance) {
        alert("Insufficient funds for this withdrawal!");
        return;
    }

    currentbalance -= withdrawalAmount;
    document.getElementById("working_area").innerHTML = "Withdrawal successful! Your updated balance is EUR" + currentbalance;

    const userIndex = balances.findIndex(entry => entry.user === currentuser);
    if (userIndex !== -1) {
        balances[userIndex].balance = currentbalance;
    } else {
        alert("User not found in balances.");
    }
}
```
- Same as deposit but subtracts withdrawal amount (-=).
- Checks sufficient funds.

7.
```
function saveUpdatedBalances(balances) {
    const dataStr = JSON.stringify(balances, null, 2);
    const blob = new Blob([dataStr], { type: "application/json" });
    const url = URL.createObjectURL(blob);

    const a = document.createElement("a");
    a.href = url;
    a.download = "balances.json";
    a.click();

    URL.revokeObjectURL(url);
}
```
- JSON.stringify(): Converts JS to JSON string.
- Blob(): Creates file-like object.
- URL.createObjectURL(): Creates URL for blob.
- Creates anchor element and triggers download.
- URL.revokeObjectURL(): Frees memory.

8.

```
function transfer() {
    document.getElementById("working_area").innerHTML = "This function is currently not available";
}
```

- Shows message that transfer is not implemented.

9.

```
function logoff() {
    alert("Please confirm you want to log off!");
    saveUpdatedBalances(balances);
    location.reload();
}
```

- alert(): Confirmation popup.
- Calls saveUpdatedBalances().
- location.reload(): Reloads page to log off.

Summary of commands/methods:

document.getElementById(): Select element by ID.
.addEventListener(): Add event listener.
e.preventDefault(): Prevent default event behavior.
fetch(): Make network request.
.then(): Handle promise success.
.catch(): Handle errors.
forEach(): Iterate array.
some(): Check if some elements meet condition.
find(): Find first element matching condition.
findIndex(): Find index of element.
innerHTML: Get/set HTML content.
style.display: Show/hide element.
prompt(): Get user input.
alert(): Show popup.
isNaN(): Check if not a number.
Number(): Convert to number.

JSON.stringify(): Convert to JSON string.

Blob(): Create file object.

URL.createObjectURL(): Create URL for file.

location.reload(): Reload webpage.

console.log(): Log message.