Copilot

Je m'excuse pour la confusion précédente. Voici le code séparé en étapes détaillées :

Étape 1: Importation de la bibliothèque Redux

```
import { createStore } from "https://cdn.skypack.dev/redux@4.0.5";
```

Étape 2: Trouver les éléments dans le document HTML

```
const score = document.getElementById("score");
const player1Button = document.getElementById("player-1");
const player2Button = document.getElementById("player-2");
const resetButton = document.getElementById("reset");
const pauseButton = document.getElementById("pause");
```

Étape 3: Ajouter des écouteurs d'événements aux boutons

```
player1Button.addEventListener("click", () => {
  store.dispatch(pointScored("player1"));
});

player2Button.addEventListener("click", () => {
  store.dispatch(pointScored("player2"));
});

resetButton.addEventListener("click", () => {
  store.dispatch(restartGame());
});

pauseButton.addEventListener("click", function () {
  store.dispatch(playPause());
});
```

Étape 4: Définir la fonction pour mettre à jour le texte du score

```
function updateScoreText(
  playing,
  winner = null,
  player1Score = 0,
  player2Score = 0,
  advantage = null
) {
  if (winner) {
    if (winner === "player1") {
      score.innerHTML = "Joueur 1 gagne";
    } else {
      score.innerHTML = "Joueur 2 gagne";
    }
  } else if (playing === false) {
    score.innerHTML = "C'est la pause";
  } else {
    let text = "Le score est: " + player1Score + " - " + player2Score;
    if (advantage) {
      if (advantage === "player1") {
        text += " avantage joueur 1";
      } else {
        text += " avantage joueur 2";
      }
    }
    score.innerHTML = text;
  }
}
```

Étape 5: Définir l'état initial

```
const initialState = {
  player1: 0,
```

```
  player2: 0,
  advantage: null,
  winner: null,
  playing: true
};
```

Étape 6: Définir les créateurs d'actions

```
const playPause = () => ({ type: "playPause" });
const restartGame = () => ({ type: "restart" });
const pointScored = (player) => ({
  type: "pointScored",
  payload: { player: player }
});
```

Étape 7: Définir le réducteur

```
function reducer(state, action) {
  if (action.type === "restart") {
    return initialState;
  }
  if (action.type === "playPause") {
    return {
      ...state,
      playing: !state.playing
    };
  }
  if (action.type === "pointScored") {
    const player = action.payload.player;
    const otherPlayer = player === "player1" ? "player2" : "player1";
    if (state.winner) {
      return state;
    }
    if (state.playing === false) {
      return state;
    }
    const currentPlayerScore = state[player];
    if (currentPlayerScore <= 15) {
      return { ...state, [player]: currentPlayerScore + 15 };
    }
    if (currentPlayerScore === 30) {
      return { ...state, [player]: 40 };
    }
    if (currentPlayerScore === 40) {
      if (state[otherPlayer] !== 40) {
        return { ...state, winner: player };
      }
      if (state.advantage === player) {
        return { ...state, winner: player };
      }
      if (state.advantage === null) {
        return { ...state, advantage: player };
      }
      return { ...state, advantage: null };
    }
  }
  return state;
}
```

Étape 8: Créer le magasin

```
const store = createStore(reducer, initialState);
```

Étape 9: Souscrire au magasin pour suivre les changements d'état

```
store.subscribe(() => {
  const state = store.getState();
  updateScoreText(
    state.playing,
    state.winner,
    state.player1,
    state.player2,
    state.advantage
```

```
  );
});
```

J’espère que cela vous aide à comprendre le code étape par étape. Si vous avez d’autres questions, n’hésitez pas à demander! 😊