```
// Variablen
//let zahler = 0; // Serveranfragen-Zähler für Tests
let anzahl = 10; // für Zeilen & Spalten -> 10 x 10 = 100 Felder
let index = 'a'; // Zeilenanfang für Schleifen
let spieleraktion = ""; // Klick des Spielers (ID des angeklicktgen Elements)
let spielerID = ""; // vom Server zugewiesen
let grund = ""; // Zweck der Abfrage an den Server:
"beitritt","beitrittsabfrage","token","spielzug"
let intervall = null; // zur Speicherung der ID des Timers
let abfrageZeit = 5000; // alle 5 Sekunden -> Timer
let eintrag = ""; // Feld ID vom Spielzug

// für Statistik
let zuganzahl = 0; // Gesamanzahl der eigenen gemachten Spielzüge
let treffer = 0; // Anzahl der Treffer der eigenen Spielzüge

// Anzahl der Schiffe (mit Feldgröße)
let schiff2 = 4; // 4 U-Boote
let schiff3 = 3; // 3 Zerstörer
let schiff4 = 2; // 2 Kreuzer
let schiff5 = 1; // 1 Schlachtschiff

let schiffe = []; // Schiffsammlung (Array)
let schifflange = []; // Schifflänge für die Schiffanzeige -> Zuordnung zu "schiffe"

// Startzustand herstellen nach Laden der Seite
window.addEventListener("load", ladenErfolgreich);
function ladenErfolgreich()
{
    document.getElementById("start").addEventListener("click", meldungPlatzierung); //
    Meldung an Spieler, da Schiffe nicht platziert
    vorbereiten("spieler", feldauswahl); // was soll vorbereitet werden? "feldauswahl" /
    "spielzug" + welches Feld
    //vorbereiten("gegner", spielzug); // Testaufruf zur Überprüfung
    vorbereitenFelder();
    document.getElementById("status").innerHTML = "Schiffe platzieren.";
}

function meldungPlatzierung() // Information für Spielstart
{
    alert("Es wurden noch nicht alle Schiffe platziert."); // Meldung an Spieler
}

function timerStarten() // Verbindungsaufbau für Spiel starten
{
    if (document.getElementById("spieler").value !== "") // Spielernamen eigegeben?
    {
        if(document.getElementById("spieler").value.includes(";"))
        {
            alert("Ungültiges Zeichen ';' im Spielernamen entdeckt!");
        }
        else
        {
            document.getElementById("spieler").readOnly = true; // Feld für Spielernamen
            sperren
            document.getElementById("start").removeEventListener("click", timerStarten); //
            nur 1x Spiel beginnen
            grund = "beitritt";
            serverAnfrage(); // Spielbeitritt
        }
    }
    else
    {
        alert("Spielernamen eingeben!"); // Meldung, dies nachzuholen
    }
}

// Spalzierung der Schiffe (feld="spieler"; auswahl=feldauswahl) / Spielzüge (feld="gegner";
auswahl=spielzug)
```

```
64  function vorbereiten(feld, auswahl) // Parameter
65  {
66      //console.log("Plazierung wird vorbereitet..."); // Testausgabe
67      index = 'a'; // Zurücksetzen des Indexes
68
69      // Für jede Reihe ( A - J )
70      for (let zahlReihe = 1; zahlReihe <= anzahl; zahlReihe++)
71      {
72          // jeweilige Reihe
73          for (let zahl = 1; zahl <= anzahl; zahl++)
74          {
75              //console.log("Index: "+index); // Testausgabe
76              let idFeld = feld + "." + index + zahl;
77              //console.log("idFeld add: " + idFeld); // id-Ausgabe zum Test
78              document.getElementById(idFeld).addEventListener("click", auswahl);
79              //console.log(auswahl); // Testausgabe
80          }
81          index = String.fromCharCode(index.charCodeAt(0) + 1); // nächster Buchstabe
82      }
83  }
84
85  // Logik der Platzierung mit Einhaltung der Regeln
86
87  function feldauswahl()
88  {
89      //console.log(this.id); // id des Elements (Testausgabe)
90      let id = this.id;
91      const schiffsflache = id.split("."); // ID aufteilen
92      let aktion = schiffsflache[1]; // Fläche-Koordinaten zuweisen
93      //console.log(aktion); // Testausgabe für das reine Feld
94
95      if (spieleraktion === "") // Speicherung des ersten Klicks
96      {
97          spieleraktion = aktion;
98      }
99      else // Logik für 2. Auswahl
100     {
101         //console.log(spieleraktion + "." + aktion);
102
103         // Überprüfung ob gerade
104         if (spieleraktion.charAt(0) === aktion.charAt(0) || spieleraktion.slice(1) === aktion.
            slice(1))
105         {
106             //console.log("Schiff gerade"); // Testausgabe
107             let lang = 0;
108             let anfang = "";
109             let richtung = "";
110
111             if (spieleraktion.charAt(0) === aktion.charAt(0)) // Zeile
112             {
113                 lang = aktion.slice(1) - spieleraktion.slice(1); // Berechnung
114                 richtung = "h"; // Anpassung der Ausrichtung
115             }
116             else // Spalte
117             {
118                 lang = spieleraktion.charCodeAt(0) - aktion.charCodeAt(0); // Berechnung
119                 richtung = "v"; // Anpassung der Ausrichtung
120             }
121             lang = korrigiereLang(lang); // Schifflänge korrigieren wegen Rechnung +
                Vorzeichen
122             //console.log("Länge: "+lang); // Testausgabe
123             //console.log("Richtung: "+richtung); // Testausgabe
124             if (6 > lang && lang > 1) // Überprüfung ob gültige Länge
125             {
126                 //let temp = null; // temporäres Zahl oder Buchstabe zur Überprüfung (Zeile /
                    Spalte)
127
128                 /* Testausgaben zur Fehlerüberprüfung der Logik
129                 //console.log("Spieleraktion: " + spieleraktion + ", Aktion: " + aktion);
```

```
130                 //console.log("Character: ");
131                 //console.log(spieleraktion.charAt(0) <= aktion.charAt(0));
132                 //console.log("Integer: ");
133                 //console.log(parseInt(spieleraktion.slice(1)) <= parseInt(aktion.slice(1)));
134                 */
135                 if (spieleraktion.charAt(0) <= aktion.charAt(0) && (parseInt(spieleraktion.
                    slice(1)) <= parseInt(aktion.slice(1))))
136                 {
137                     //console.log("if"); // Testausgabe
138                     anfang = spieleraktion; // erste Aktion
139                 }
140                 else
141                 {
142                     //console.log("else"); // Testausgabe
143                     anfang = aktion; // zweite Aktion
144                 }
145 
146                 //console.log("Anfang: " + anfang); // Testausgabe
147                 platziereSchiff(anfang, lang, richtung); // Übergabe der benötigten Werte
148             }
149             else
150             {
151                 alert("Gewählte Schiffgröße existiert nicht!\nLänge: " + lang); // Meldung an
                    Spieler
152             }
153         }
154         else
155         {
156             alert("Schiffe können nicht diagonal platziert werden!"); // Meldung an Spieler
157         }
158 
159         spieleraktion = ""; // Spieleraktion wird zurückgesetzt (1. Klick)
160     }
161 }
162 
163 // Korrektur der berechneten Schiffslänge
164 
165 function korrigiereLang(wert)
166 {
167     if (wert < 0) // Wert negativ
168     {
169         wert /= -1; // Vorzeichenumkehr zu +
170     }
171     return ++wert; // +1 für Korrektur
172 }
173 
174 // Platzierung der Schiffe auf dem Spielfeld mit Sperrflächen
175 
176 function platziereSchiff(start, wert, ausrichtung)
177 {
178     // start = Schleifenanfang (links oben), wert = Länge, ausrichtung vertikal / horizontal
        (v/h)
179     //console.log("Start: " + start + ", Wert: " + wert + ", Ausrichtung: " + ausrichtung);
        // Testausgabe zur Überprüfung der Werte
180 
181     let schifftyp = "";
182     let schiffVorhanden = false;
183     let schiffsflache = []; // Vorlage zur Reservierung des benötigten Bereichs für das Schiff
184     let sperrbereich = []; // Vorlage zur Sperrung der Bereiche nebenan
185     //console.log("Ausrichtung: " + ausrichtung); // Testausgabe
186 
187     switch (wert) // Schifftyp festlegen + Prüfung ob vorhanden (Bestand > 0)
188     {
189         case 5:
190             schifftyp = "Schlachtschiff";
191             if (schiff5 > 0)
192             {
193                 schiffVorhanden = true;
194             }
```

```
195             break;
196         case 4:
197             schifftyp = "Kreuzer";
198             if (schiff4 > 0)
199             {
200                 schiffVorhanden = true;
201             }
202             break;
203         case 3:
204             schifftyp = "Zerstörer";
205             if (schiff3 > 0)
206             {
207                 schiffVorhanden = true;
208             }
209             break;
210         case 2:
211             schifftyp = "U-Boot";
212             if (schiff2 > 0)
213             {
214                 schiffVorhanden = true;
215             }
216     }
217
218     if (schiffVorhanden) // wenn Schifftyp Anzahl > 0
219     {
220         // benötigte Elemente in schiffsflache laden + sperrbereich neben Schiff befüllen
221         for (let zeichne = 0; wert > zeichne; zeichne++)
222         {
223             let IDsperreDavor = "";
224             let IDsperreDanach = "";
225             //console.log("Start: " + start); // Testausgabe
226             let IDzusatz = "";
227             if (ausrichtung === "h") // horizontal
228             {
229                 let temp = zeichne + parseInt(start.slice(1)); // zur vorbereitung der
                    variablen Zahl der ID
230                 //console.log("Temp: " + temp); // Testausgabe
231                 IDzusatz = start.charAt(0) + temp; // für die Fläche des Schiffs
232                 //console.log("IDzusatz: " + IDzusatz);
233                 IDsperreDavor = String.fromCharCode(start.charCodeAt(0) - 1) + temp;
234                 IDsperreDanach = String.fromCharCode(start.charCodeAt(0) + 1) + temp;
235                 //console.log("von " + IDsperreDavor + " bis " + IDsperreDanach); //
                    Testausgabe für davor und danach sperren
236             }
237             else // vertikal
238             {
239                 let temp = String.fromCharCode(zeichne + start.charCodeAt(0)); // zur
                    vorbereitung des variablen Buchstabens der ID
240                 //console.log("Temp: " + temp); // Testausgabe
241                 IDzusatz = temp + start.slice(1); // für die Fläche des Schiffs
242                 //console.log("IDzusatz: " + IDzusatz);
243                 IDsperreDavor = temp + (start.slice(1) - 1);
244                 IDsperreDanach = temp + (parseInt(start.slice(1)) + 1);
245                 //console.log("von " + IDsperreDavor + " bis " + IDsperreDanach); //
                    Testausgabe für davor und danach sperren
246             }
247             //console.log("schiffsflache..."); // Testausgabe
248
249             // nur existierende Elemente in Array schreiben
250             let pruefen = document.getElementById("spieler." + IDsperreDavor);
251             if (pruefen !== null)
252             {
253                 sperrbereich.push(pruefen);
254             }
255             pruefen = document.getElementById("spieler." + IDsperreDanach);
256             if (pruefen !== null)
257             {
258                 sperrbereich.push(pruefen);
259             }
```

```
260             schiffsflache.push(document.getElementById("spieler." + IDzusatz));
261         }
262 
263         let schiffanfang = schiffsflache[0].id.split("."); // Sperrelement für Start
264         let schiffende = schiffsflache[schiffsflache.length - 1].id.split("."); //
            Sperrelement für Ende
265 
266         // Sperrbereich erweitern: eins nach vorne + eins nach hinten (Schiff), falls möglich
            (document.getElementById(x) == null, wenn nicht da)
267 
268         if (ausrichtung === "h") // horizontale Ausrichtung
269         {
270             // Verschiebung nach vorne bzw. hinten (vom Schiff aus) -> Anpassung des fixen
                Wertes (Zahl)
271             let zusatzsperreAnfang = schiffanfang[1].slice(1) - 1; // Zahl von
272             let zusatzsperreEnde = parseInt(schiffende[1].slice(1)) + 1; // Zahl bis
273             //console.log("ZusatzsperreAnfang: " + zusatzsperreAnfang); // Testausgabe
274             //console.log("ZusatzsperreEnde: " + zusatzsperreEnde); // Testausgabe
275 
276             for (let zusatz = -1; zusatz < 2; zusatz++) // Verschiebung nach oben -> -1
277             {
278                 // linke Seite sperren
279                 let temp = String.fromCharCode(schiffanfang[1].charCodeAt(0) + zusatz) +
                    zusatzsperreAnfang; // Erzeugung der ID links
280                 //console.log("ID: " + "spieler." + temp); // Testausgabe
281                 let pruefen = document.getElementById("spieler." + temp);
282                 if (pruefen !== null)
283                 {
284                     sperrbereich.push(pruefen);
285                     //console.log("Test: "+pruefen); // Testausgabe
286                 }
287                 pruefen = null; // zurücksetzen
288 
289                 // rechte Seite sperren
290                 temp = String.fromCharCode(schiffende[1].charCodeAt(0) + zusatz) +
                    zusatzsperreEnde; // Erzeugung der ID rechts
291                 pruefen = document.getElementById("spieler." + temp);
292                 //console.log("neu Temp: " + temp); // Testausgabe
293                 if (pruefen !== null)
294                 {
295                     sperrbereich.push(pruefen);
296                     //console.log("Test: " + pruefen.id); // Testausgabe
297                 }
298             }
299         }
300         else // vertikale Ausrichtung
301         {
302             // Verschiebung nach vorne bzw. hinten (vom Schiff aus) -> Anpassung des fixen
                Wertes (Buchstabe)
303             let zusatzsperreAnfang = String.fromCharCode(schiffanfang[1].charCodeAt(0) - 1);
                // Buchstabe von
304             let zusatzsperreEnde = String.fromCharCode(schiffende[1].charCodeAt(0) + 1); //
                Buchstabe bis
305             //console.log("ZusatzsperreAnfang: " + zusatzsperreAnfang); // Testausgabe
306             //console.log("ZusatzsperreEnde: " + zusatzsperreEnde); // Testausgabe
307 
308             for (let zusatz = -1; zusatz < 2; zusatz++) // Verschiebung nach links -> -1
309             {
310                 // linke Seite sperren
311                 let temp = zusatzsperreAnfang + (parseInt(schiffanfang[1].slice(1)) + zusatz);
                     // Erzeugung der ID oben
312                 //console.log("ID: " + "spieler." + temp); // Testausgabe
313                 let pruefen = document.getElementById("spieler." + temp);
314                 if (pruefen !== null)
315                 {
316                     sperrbereich.push(pruefen);
317                     //console.log("Test: "+pruefen); // Testausgabe
318                 }
319                 pruefen = null; // zurücksetzen
```

```
320
321                 // rechte Seite sperren
322                 temp = zusatzsperreEnde + (parseInt(schiffende[1].slice(1)) + zusatz); //
                    Erzeugung der ID unten
323                 pruefen = document.getElementById("spieler." + temp);
324                 //console.log("neu Temp: " + temp); // Testausgabe
325                 if (pruefen !== null)
326                 {
327                     sperrbereich.push(pruefen);
328                     //console.log("Test: " + pruefen.id); // Testausgabe
329                 }
330                 //console.log("------------"); // Testausgabe (Trennung der
                    Schleifendurchläufen)
331             }
332         }
333
334         let check = true; // alle Felder müssen frei sein
335         for (let element of schiffsflache) // Prüfen ob Felder frei für Platzierung
336         {
337             //console.log(element); // Testausgabe
338             if (element.innerHTML === "X" || element.innerHTML === "O")
339             {
340                 check = false; // unfreies Feld gefunden
341                 break; // Schleifenabbruch
342             }
343         }
344
345         if (check) // wenn frei
346         {
347             switch (wert) // Schiff entnehmen für Platzierung & Schiffanzeige aktualisieren
348             {
349                 case 5:
350                     schiff5--;
351                     document.getElementById("schiff5").innerHTML = parseInt(document.
                        getElementById("schiff5").innerHTML) + 1;
352                     break;
353                 case 4:
354                     schiff4--;
355                     document.getElementById("schiff4").innerHTML = parseInt(document.
                        getElementById("schiff4").innerHTML) + 1;
356                     break;
357                 case 3:
358                     schiff3--;
359                     document.getElementById("schiff3").innerHTML = parseInt(document.
                        getElementById("schiff3").innerHTML) + 1;
360                     break;
361                 case 2:
362                     schiff2--;
363                     document.getElementById("schiff2").innerHTML = parseInt(document.
                        getElementById("schiff2").innerHTML) + 1;
364             }
365
366             let array = []; // für IDs aller Flächen eines Schiffes ohne "spieler." Zusatz
367
368             for (const element of schiffsflache) // Schiff platzeren
369             {
370                 element.innerHTML = "O";
371                 element.style.backgroundColor = "grey";
372                 let temp = element.id.split("."); // Spieler und Flächenkoordinaten trennen
373                 //console.log(temp[1]); // Testausgabe
374                 array.push(temp[1]);
375             }
376             document.getElementById("status").innerHTML = schifftyp + " wurde platziert."; //
                Statusausgabe
377             schiffe.push(array); // Schiffe im Array sammeln
378             schifflange.push(wert);
379             //console.log(schiffe); // Testausgabe
380
381             // Bereich drumherum sperren
```

```
382             for (const element of sperrbereich) // Sperrfläche um das Schiff drum herum
                platzieren
383             {
384                 element.innerHTML = "X";
385                 element.style.backgroundColor = "yellow";
386             }
387             //console.log(sperrbereich); // Testausgabe des Sperrbereichs
388 
389             if (schiff5 === 0 && schiff4 === 0 && schiff3 === 0 && schiff2 === 0) // Abfrage
                ob Schiffe übrig zum platzieren
390             {
391                 wasser(); // alle Schiffe wurden platziert
392             }
393 
394         }
395         else // mindestens ein Feld belegt
396         {
397             document.getElementById("status").innerHTML = schifftyp + " konnte aufgrund einer
                Kollision nicht platziert werden!";
398         }
399 
400         //console.log(check); // Testausgabe
401     }
402     else
403     {
404         document.getElementById("status").innerHTML = "Kein " + schifftyp + " mehr verfügbar!"
            ;
405     }
406 }
407 
408 function wasser() // wandelt Sperrflächen zu Wasserflächen um & entfernt die Evenlistener für
    die Platzierung der Schiffe
409 {
410     //console.log("Wasser wird verschüttet..."); // Testausgabe
411     index = 'a'; // Zurücksetzen des Indexes
412 
413     // Für jede Reihe ( A - J )
414     for (let zahlReihe = 1; zahlReihe <= anzahl; zahlReihe++)
415     {
416         // jeweilige Reihe
417         for (let zahl = 1; zahl <= anzahl; zahl++)
418         {
419             //console.log("Index: "+index); // Testausgabe
420             let idFeld = "spieler." + index + zahl;
421             //console.log(idFeld); // id-Ausgabe zum Test
422             if (document.getElementById(idFeld).innerHTML !== "O") // kein Schiff platziert
423             {
424                 document.getElementById(idFeld).innerHTML = "W"; // Wasserzeichen
425                 document.getElementById(idFeld).style.backgroundColor = "aqua"; //
                    Wasserhintergrundfarbe
426             }
427             //console.log("idFeld remove: " + idFeld);
428             document.getElementById(idFeld).removeEventListener("click", feldauswahl); //
                entferne die Auswahlmöglichkeit auf dem eigenen Spielfeld
429         }
430         index = String.fromCharCode(index.charCodeAt(0) + 1); // nächster Buchstabe
431     }
432     setTimeout(meldungNachPlatzierung, 200); // damit Meldung erst nach den platzieren des
    letzten Schiffs auftaucht (200ms Verzögerung)
433 }
434 
435 function meldungNachPlatzierung() // gibt die Bestätigung für den Spieler aus, dass alle
    Schiffe platziert wurden.
436 {
437     alert("Es wurden alle Schiffe platziert.");
438     document.getElementById("start").removeEventListener("click", meldungPlatzierung); //
    Hinweis zur Platzierung der Schiffe entfernen
439     document.getElementById("start").addEventListener("click", timerStarten); // Spiel bereit
    zum Starten -> Verbinden mit Server
```

```
}

function vorbereitenFelder() // alle Spielflächen beider Spielfelder mit Zeichen vorbelegen
{
    //console.log("Wasser wird verschüttet..."); // Testausgabe
    index = 'a'; // Zurücksetzen des Indexes

    // Für jede Reihe ( A - J )
    for (let zahlReihe = 1; zahlReihe <= anzahl; zahlReihe++)
    {
        // jeweilige Reihe
        for (let zahl = 1; zahl <= anzahl; zahl++)
        {
            //console.log("Index: "+index); // Testausgabe
            let idFeldSpieler = "spieler." + index + zahl;
            let idFeldGegner = "gegner." + index + zahl;
            //console.log(idFeld); // id-Ausgabe zum Test
            document.getElementById(idFeldSpieler).innerHTML = "W"; // Wasserzeichen
            document.getElementById(idFeldGegner).innerHTML = "?"; // unbekannte Gegnerflächen
            //console.log("idFeld remove: " + idFeld); // Testausgabe
        }
        index = String.fromCharCode(index.charCodeAt(0) + 1); // nächster Buchstabe
    }
}

// Spielersuche zum Registrieren in Datei zum Aufbau des Spiels
function serverAnfrage()
{
    //zahler++; // AnfrageNr für Tests
    //console.log("Serveranfrage: " + zahler); // Testausgabe

    // Anfang - Browserweiche
    try
    {
        // Eintrag von Verbindungsdaten in xhttp
        xhttp = new XMLHttpRequest(); // Firefox
    }
    catch (error)
    {
        try
        {
            xhttp = new ActiveXObject("Msxml2.XMLHTTP"); // Microsoft
        }
        catch (error)
        {
            try
            {
                xhttp = new ActiveXObject("Microsoft.XMLHTTP");
            }
            catch (error)
            {
                return;
            }
        }
    }
    // Ende - Browserweiche

    // liest ständig die Rückmeldungen
    xhttp.onreadystatechange = ajax;

    // Verbindung wird geöffnet mit method="POST"
    xhttp.open("POST", "script.php");

    //console.log("Grund der Serveranfrage: " + grund); // Testausgabe

    if (grund === "beitritt") // Abfrage für Grund
    {
        verbinden(); // Versuch sich für Spiel zu registrieren
    }
```

```
509         else if (grund === "beitrittsabfrage")
510         {
511             beitrittsabfrage(); // Prüfen ob Spieler 2 beigetreten
512         }
513         else if (grund === "token")
514         {
515             //console.log("Form für Token???"); // Testausgabe
516             tokenAbfrage();
517         }
518         else if (grund === "spielzug") // bei Schuss auf Feld
519         {
520             sendeSpielzug();
521         }
522     }
523
524     function ajax() // Antwort des php-Skripts
525     {
526         // readyState = 4 -> Rückmeldung vollständig ; status = 200 -> Rückmeldung fehlerfrei
527         if (this.readyState == 4 && this.status == 200)
528         {
529             let antwort = this.responseText;
530             //console.log("Testantwort: " + antwort); // Testausgabe
531             // function für weiteren Ablauf
532             if (grund === "beitritt")
533             {
534                 antwortbearbeitungBeitrittsanfrage(antwort); // Spieler will sich registrieren
535             }
536             else if (grund === "beitrittsabfrage")
537             {
538                 anwortbearbeitungBeitrittsabfrage(antwort); // Spieler1 fragt nach Spieler2
539             }
540             else if (grund === "token")
541             {
542                 antwortbearbeitungTokenAbfrage(antwort);
543                 //console.log("Hab ich Token?"); // Testausgabe
544             }
545             else if (grund === "spielzug") // Rückmeldung des Schusses auf ein Feld
546             {
547                 antwortbearbeitungSendeSpielzug(antwort);
548             }
549         }
550     }
551
552     function antwortbearbeitungBeitrittsanfrage(antwort) // Anfrageverarbeitung Beitritt
553     {
554         // gesplittetes Array. Übergabe: Spieler1 registriert. oder Spieler2 registriert. Gegner:
        [Spieler2 Name]
555         let check = antwort.split(";"); // Trennzeichen ";" -> als Spielername unzulässig
556         spielerID = check[0];
557         //console.log(spielerID); // Testausgabe
558         //console.log("antwortbearbeitungBeitrittsanfrage"); // Testausgabe
559         if (check[1] === "registriert.") // Registrierung für Spiel erfolgreich
560         {
561             vorbereiten("gegner", spielzug); // Spielzug bereit machen
562             //console.log("Check: "+check); // Testausgabe
563
564             if (check.length > 2) // Registrierung für Spieler2 erfolgt
565             {
566                 //console.log("Check: " + check); // Testausgabe
567                 //console.log("Checkelement: " + check[check.length - 3]); // Testausgabe
568                 document.getElementById("gegner").value = check[check.length - 3]; // Spieler2
                    eintragen
569                 alert("Spieler " + check[check.length - 3] + " ist dem Spiel beigetreten."); //
                    Statusmeldung
570                 // direkt Abfrage, ob er dran ist..... -> token
571                 //console.log("Token: " + check[check.length - 2]); // Testausgabe des Tokens, ob
                    Spieler1 oder Spieler2
572                 if (spielerID === check[check.length - 2]) // ich (Spieler2) hab Token
573                 {
```

```
574                 //console.log("Ich hab Token."); // Testausgabe
575                 grund = "spielzug";
576                 document.getElementById("status").innerHTML = "Spielzug erwartet."; //
                    Statusmeldung -> Spieler ist dran
577             }
578             else // Timer für Abfrage ob Token
579             {
580                 grund = "token";
581                 intervall = setInterval(serverAnfrage, abfrageZeit);
582                 document.getElementById("status").innerHTML = "Warten auf gegnerischen
                    Spielzug..."; // Statusmeldung -> Spieler ist nicht dran
583             }
584 
585         }
586         else // regelmäßige Abfrage, ob Gegner 2 beigetreten ist in Gang setzen
587         {
588             //console.log("antwortbearbeitungBeitrittsanfrage: " + zahler); // Testausgabe
589             grund = "beitrittsabfrage";
590             intervall = setInterval(serverAnfrage, abfrageZeit); // alle 5 Sekunden
591         }
592     }
593     else // Registrierung fehlgeschlagen
594     {
595         document.getElementById("status").innerHTML = "Spielbeitritt nicht möglich.";
596         alert(antwort); // Meldung: Spiel bereits im Gange.
597     }
598 }
599 
600 function verbinden() // Spielbeitrittsversuch
601 {
602     document.getElementById("status").innerHTML = "Spielbeitritt erfolgt...";
603 
604     // Vorlage für Formular erstellen + Daten mit "append" anheften
605     let formDaten = new FormData();
606 
607     let jsonString = JSON.stringify(schiffe); // JSON String generieren
608     formDaten.append("spieler", document.getElementById("spieler").value);
609     formDaten.append("aufstellung", jsonString); // JSON String mitschicken
610     //console.log(jsonString); // Testausgabe
611 
612     // Anfrage wird gesendet
613     xhttp.send(formDaten);
614 }
615 
616 function beitrittsabfrage() // alle 5 Sekunden bis Spieler2 beigetreten
617 {
618     // Anfrage wird gesendet
619     xhttp.send(); // keine Form angehängt
620 }
621 
622 function anwortbearbeitungBeitrittsabfrage(antwort) // Überprüfung ob Spieler2 beigetreten ist
623 {
624     //console.log("Beitritt: " + antwort); // Testausgabe
625     if (antwort !== "") // nur wenn Spieler2 eingetragen ist
626     {
627         document.getElementById("status").innerHTML = "Warten auf Spielzug...";
628         document.getElementById("gegner").value = antwort; // Gegner für Spieler1 eintragen
629         grund = "token"; // ob Spieler dran ist
630         //clearInterval(intervall); // nur wenn Spieler dran ist....
631     }
632 }
633 
634 function tokenAbfrage() // Sende Anfrage an Token
635 {
636     let formDaten = new FormData();
637 
638     formDaten.append("spielerID", spielerID); // übergeben ID
639 
640     // Anfrage wird gesendet
```

```
641         xhttp.send(formDaten);
642     }
643 
644     function antwortbearbeitungTokenAbfrage(antwort) // Überprüfung, ob eigener Token
645     {
646         let temp = antwort.split(";"); // Seperatisieren von Token & letzter Spielzug
647         //console.log("Token: " + temp[0]); // Testausgabe
648         //console.log("ID Feld: " + temp[1]); // Testausgabe
649         if (temp[0] === spielerID) // Eigener Token
650         {
651             clearInterval(intervall); // Intervall entfernen, da Spieler mit Spielzug dran ist
652             grund = "spielzug"; // nächste serverAnfrage hat den grund für spielzug
653 
654             //console.log("A"+temp[1]+"B"); // Testausgabe
655             if (temp[1] !== "") // Abfangen vom Erstzug von Spieler1
656             {
657                 let position = -1; // für das versunkene Schiff im Array der "schifflange"
658 
659                 //console.log("Schiffe: " + schiffe); // Testausgabe
660                 for (let schiff in schiffe) // foreach JS Version
661                 {
662                     // "schiff" wird mit Koordinate gefiltert, falls vorhanden
663                     //console.log("Schiffindex: "+schiffe.indexOf(schiff) + " hat die Länge " +
                        schiffe[schiff].length); // Testausgabe
664                     if (schiffe[schiff].includes(temp[1])) // Überprüfung ob gegnerischer
                        Spielzug trifft
665                     {
666                         schiffe[schiff] = schiffe[schiff].filter(e => e !== temp[1]); //
                            entfernen der beschossenen Fläche
667                         if (schiffe[schiff].length === 0) // keine beschießbaren Flächen mehr
668                         {
669                             // Treffer merken welches Schiff !!!
670                             position = schiff; // setzen der Position des versunkenen Schiffs
671                             //console.log("Eigenes Schiff verloren."); // Testausgabe
672                         }
673                         //console.log("Schiff: "); // Testausgabe
674                         //console.log(schiffe[schiff]); // Testausgabe
675                     }
676                 }
677 
678                 /* Testausgaben
679                 //console.log("Schiffe: ");
680                 //console.log(schiffe);
681                 //console.log("Position (Array): " + position);
682                 //console.log("Länge des Schiffes: " + schifflange[position]);
683                 */
684 
685                 if (-1 < position) // wenn Position gesetzt (Schiff versunken)
686                 {
687                     switch (schifflange[position])  // Meldung über Art des Schiffes +
                        Aktualisierung der Schiffanzeige
688                     {
689                         case 2:
690                             document.getElementById("status").innerHTML = "Eigenes U-Boot wurde
                                versenkt!";
691                             document.getElementById("schiff2").innerHTML = parseInt(document.
                                getElementById("schiff2").innerHTML) - 1;
692                             break;
693                         case 3:
694                             document.getElementById("status").innerHTML = "Eigener Zerstörer
                                wurde versenkt!";
695                             document.getElementById("schiff3").innerHTML = parseInt(document.
                                getElementById("schiff3").innerHTML) - 1;
696                             break;
697                         case 4:
698                             document.getElementById("status").innerHTML = "Eigener Kreuzer wurde
                                versenkt!";
699                             document.getElementById("schiff4").innerHTML = parseInt(document.
                                getElementById("schiff4").innerHTML) - 1;
```

```
                        break;
                    case 5:
                        document.getElementById("status").innerHTML = "Eigenes Schlachtschiff
                        wurde versenkt!";
                        document.getElementById("schiff5").innerHTML = parseInt(document.
                        getElementById("schiff5").innerHTML) - 1;
                }
            }

            //console.log("Schifflänge: " + schifflange); // Testausgabe
            //console.log("Schiff verloren: " + schifflange[index]); // Testausgabe

            // gegnerischer Spielzug verarbeiten
            if (document.getElementById("spieler." + temp[1]).innerHTML === "W")
            {
                document.getElementById("spieler." + temp[1]).style.backgroundColor = "yellow"
                ; // wenn Schuss auf Wasser, dann Gelb
                document.getElementById("status").innerHTML = "Gegnerischer Schuss ging ins
                Wasser."; // Meldung, dass Schuss daneben
            }
            else if (document.getElementById("spieler." + temp[1]).innerHTML === "O")
            {
                document.getElementById("spieler." + temp[1]).style.backgroundColor = "red";
                // wenn Treffer eines Schiffes, dann Rot
            }
            document.getElementById("spieler." + temp[1]).innerHTML = "X"; // Treffer
            markieren auf eigenem Spielfeld
        }
        if (document.getElementById("schiff2").innerHTML === "0" && document.getElementById(
        "schiff3").innerHTML === "0" && document.getElementById("schiff4").innerHTML === "0"
        && document.getElementById("schiff5").innerHTML === "0")
        {
            aktionEntfernen(); // Entfernen des weiteren Beschusses
            statistikausgabe(); // Statistik unter Schiffanzeige platzieren
            setTimeout(verlorenMeldung, 200); // Meldung über Niederlage (200 ms Verzögerung)
        }
        else
        {
            setTimeout(erwarten, abfrageZeit); // verzögerte Statusmeldung, dass Spielzug
            erfolgen soll (5 Sekunden Verzögerung)
        }
    }
}

function verlorenMeldung() // verzögerte Meldung, dass verloren
{
    document.getElementById("status").innerHTML = "Alle Schiffe verloren!";
    alert("Spiel verloren!");
}

function erwarten() // Statusmeldung
{
    document.getElementById("status").innerHTML = "Spielzug erwartet."; // Statusmeldung,
    dass Spieler Zug tätigen soll
}

// Logik der Spielzüge

function spielzug() // beim Klick auf gegnerisches Spielfeld
{
    if (document.getElementById(this.id).innerHTML === "?") // unbekanntes Feld
    {
        let id = this.id;
        //console.log(id); // Testausgabe
        if (grund === "spielzug") // nur wenn Spieler dran ist
        {
            zuganzahl++;
            let temp = id.split("."); // gegnerID aufteilen
            eintrag = temp[1]; // Feldkoordinate des gegnerischen Spielfeldes (ID)
```

```
759             document.getElementById(id).innerHTML = "X";
760             serverAnfrage();
761             intervall = setInterval(serverAnfrage, abfrageZeit);
762         }
763         else // wartet noch auf eigenen Token
764         {
765             alert("Gegner ist dran!"); // Meldung an Spieler
766         }
767     }
768     else
769     {
770         alert("Feld bereits beschossen!"); // Meldung an Spieler
771     }
772 }
773
774 function sendeSpielzug()
775 {
776     //console.log("Spielzug wird gesendet."); // Testausgabe
777     let formDaten = new FormData();
778
779     formDaten.append("spielerID", spielerID); // übergeben ID
780     formDaten.append("spielzug", eintrag); // übergeben Spielzug
781
782     // Anfrage wird gesendet
783     xhttp.send(formDaten);
784 }
785
786 function antwortbearbeitungSendeSpielzug(antwort) // Rückmeldung, ob (W)asser, (T)reffer,
    (V)ersenkt oder sogar (G)ewonnen
787 {
788     treffer++; // Treffer erhöhen -> Code sparen
789     switch (antwort)
790     {
791         case "W":
792             document.getElementById("status").innerHTML = "Schuss ging ins Wasser."; //
                Statusmeldung -> (W)asser!
793             document.getElementById("gegner." + eintrag).innerHTML = "W"; // Wasserzeichen
794             document.getElementById("gegner." + eintrag).style.backgroundColor = "aqua"; //
                Wasserfarbe
795             treffer--; // Treffer reduzieren, weil daneben
796             break;
797         case "T":
798             document.getElementById("status").innerHTML = "Gegnerisches Schiff getroffen!";
                // Statusmeldung -> (T)reffer!
799             document.getElementById("gegner." + eintrag).innerHTML = "T"; // Trefferzeichen
800             document.getElementById("gegner." + eintrag).style.backgroundColor = "red"; //
                Treffermarkierung
801             break;
802         case "V":
803             document.getElementById("status").innerHTML = "Gegnerisches Schiff versenkt!"; //
                Statusmeldung -> (V)ersenkt!
804             document.getElementById("gegner." + eintrag).innerHTML = "V"; // Versenktzeichen
805             document.getElementById("gegner." + eintrag).style.backgroundColor = "red"; //
                Treffermarkierung
806             break;
807         case "G":
808             document.getElementById("status").innerHTML = "Alle gegnerischen Schiffe wurden
                versenkt!"; // Statusmeldung -> (G)ewonnen!
809             document.getElementById("gegner." + eintrag).innerHTML = "V"; // Versenktzeichen
810             document.getElementById("gegner." + eintrag).style.backgroundColor = "red"; //
                Treffermarkierung
811             aufdecken(); // restliche Felder mit Wasser befüllen
812             aktionEntfernen(); // Eventlistener entfernen vom gegnerischen Spielfeld
813             statistikausgabe(); // Statistik unter Schiffanzeige platzieren
814             clearInterval(intervall); // Stop der Anfragen, da Spiel vorbei.
815             setTimeout(meldungSieg, 200); // damit Meldung erst nach der Eintragung auftritt
                (200ms Verzögerung)
816         //console.log("Spielende."); // Testausgabe
817     }
```

```
818         if (antwort !== "G") // wenn Spiel weitergeht
819         {
820             // Rückmeldung des Servers bezüglich (W)asser / (T)reffer / (V)ersenkt / (G)ewonnen
                verarbeiten
821             grund = "token"; // danach die nächste serverAnfrage nach token
822             setTimeout(meldungWarten, abfrageZeit); // damit Statuseldung erst nach der
                Statusmeldung des Spielzuges auftritt (5000ms Verzögerung)
823 
824             //console.log("antwortbearbeitungSendeSpielzug: " + antwort); // Testausgabe
825         }
826     }
827 
828     function meldungSieg() // Verzögerte Meldung des Sieges damit Markierung zuerst erfolgt
829     {
830         alert("Spiel gewonnen!");
831     }
832 
833     function meldungWarten()
834     {
835         document.getElementById("status").innerHTML = "Warten auf gegnerischen Spielzug..."; //
            Statusmeldung -> Spieler ist nicht dran
836     }
837 
838     function aufdecken() // alle restlichen Wasserfelder anzeigen (Spielfeld -> Gegner)
839     {
840         //console.log("Wasser wird verschüttet..."); // Testausgabe
841         index = 'a'; // Zurücksetzen des Indexes
842 
843         // Für jede Reihe ( A - J )
844         for (let zahlReihe = 1; zahlReihe <= anzahl; zahlReihe++)
845         {
846             // jeweilige Reihe
847             for (let zahl = 1; zahl <= anzahl; zahl++)
848             {
849                 //console.log("Index: "+index); // Testausgabe
850                 let idFeld = "gegner." + index + zahl;
851                 //console.log(idFeld); // id-Ausgabe zum Test
852                 if (document.getElementById(idFeld).innerHTML === "?")
853                 {
854                     document.getElementById(idFeld).innerHTML = "W"; // Wasserzeichen
855                     document.getElementById(idFeld).style.backgroundColor = "aqua";
856                 }
857             }
858             index = String.fromCharCode(index.charCodeAt(0) + 1); // nächster Buchstabe
859         }
860     }
861 
862     function aktionEntfernen() // Eventlistener für gegnerisches Spielfeld entfernen
863     {
864         //console.log("Spieleraktionen beenden..."); // Testausgabe
865         index = 'a'; // Zurücksetzen des Indexes
866 
867         // Für jede Reihe ( A - J )
868         for (let zahlReihe = 1; zahlReihe <= anzahl; zahlReihe++)
869         {
870             // jeweilige Reihe
871             for (let zahl = 1; zahl <= anzahl; zahl++)
872             {
873                 //console.log("Index: "+index); // Testausgabe
874                 let idFeld = "gegner." + index + zahl;
875                 //console.log("idFeld remove: " + idFeld); // id-Ausgabe zum Test
876                 document.getElementById(idFeld).removeEventListener("click", spielzug);
877                 //console.log(auswahl); // Testausgabe
878             }
879             index = String.fromCharCode(index.charCodeAt(0) + 1); // nächster Buchstabe
880         }
881     }
882 
883     function statistikausgabe() // Eintrag in Statistikbereich unter der Schiffanzeige
```

```
884  {
885      let ausgabe = "<br><br>" + treffer + " Treffer von " + zuganzahl + " Schüssen<br>"; // x
     Treffer von y Schüssen
886      ausgabe += "<br>Eigene Trefferquote: " + parseInt(100 * treffer / zuganzahl) + " %"; //
     ganze % ohne Kommastellen
887
888      document.getElementById("statistik").innerHTML = ausgabe;
889  }
890
```