# Inhaltsverzeichnis

<!DOCTYPE html> <html lang=„de“> <head>

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Gehörtraining: Tonhöhenvergleich</title>
<style>
    .ear-training-app {
        font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
        max-width: 800px;
        margin: 0 auto;
        padding: 20px;
        background-color: #f5f5f5;
    }
    .ear-training-app h1, .ear-training-app h2 {
        color: #333;
        text-align: center;
    }
    .ear-training-app .container {
        background-color: white;
        border-radius: 10px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        padding: 20px;
        margin-bottom: 20px;
    }
    .ear-training-app .settings {
        display: flex;
        flex-direction: column;
        gap: 15px;
    }
    .ear-training-app .difficulty-options {
        display: flex;
        gap: 10px;
        justify-content: center;
        margin: 10px 0;
    }
    .ear-training-app .difficulty-btn {
        padding: 10px 15px;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        font-weight: bold;
        transition: background-color 0.3s;
    }
    .ear-training-app .difficulty-btn.active {
        background-color: #4CAF50;
        color: white;
    }
    .ear-training-app .start-btn {
        background-color: #4CAF50;
        color: white;
        padding: 12px 20px;
```

```
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
            font-weight: bold;
            margin: 10px auto;
            display: block;
        }
        .ear-training-app .play-tones-btn {
            background-color: #2196F3;
            color: white;
            padding: 12px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
            font-weight: bold;
            margin: 10px auto;
            display: block;
        }
        .ear-training-app .answer-btns {
            display: flex;
            justify-content: center;
            gap: 20px;
            margin: 20px 0;
        }
        .ear-training-app .answer-btn {
            padding: 15px 30px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
            font-weight: bold;
            transition: background-color 0.3s;
        }
        .ear-training-app .higher-btn {
            background-color: #FFEB3B;
        }
        .ear-training-app .lower-btn {
            background-color: #FFEB3B;
        }
        .ear-training-app .correct {
            background-color: #4CAF50 !important;
            color: white;
        }
        .ear-training-app .incorrect {
            background-color: #F44336 !important;
            color: white;
        }
        .ear-training-app .progress {
            display: flex;
```

```
        justify-content: space-between;
        margin-top: 20px;
    }
    .ear-training-app .progress-dots {
        display: flex;
        gap: 10px;
        margin-bottom: 10px;
    }
    .ear-training-app .dot {
        width: 20px;
        height: 20px;
        border-radius: 50%;
        background-color: #ddd;
    }
    .ear-training-app .dot.correct {
        background-color: #4CAF50;
    }
    .ear-training-app .dot.incorrect {
        background-color: #F44336;
    }
    .ear-training-app .dot.current {
        border: 2px solid #2196F3;
    }
    .ear-training-app .stats {
        text-align: center;
        font-weight: bold;
        margin-top: 10px;
    }
    .ear-training-app .hidden {
        display: none;
    }
    .ear-training-app .cancel-btn {
        background-color: #9E9E9E;
        color: white;
        padding: 10px 15px;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        font-weight: bold;
        margin-top: 10px;
        display: block;
        margin: 10px auto;
    }
    .ear-training-app .result-message {
        text-align: center;
        font-size: 18px;
        margin-top: 20px;
        font-weight: bold;
    }
    .ear-training-app .audio-status {
        text-align: center;
```

```
            margin: 10px 0;
            padding: 10px;
            background-color: #FFF9C4;
            border-radius: 5px;
        }
        .ear-training-app .difficulty-description {
            font-size: 14px;
            text-align: center;
            margin-top: 5px;
            color: #666;
        }
        .ear-training-app .next-btn {
            background-color: #FF9800;
            color: white;
            padding: 10px 15px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-weight: bold;
            margin: 10px 5px;
            display: inline-block;
        }
        .ear-training-app #retry-next {
            text-align: center;
        }
    </style>
```

</head> <body>

```
    <div class="ear-training-app">
        <h1>Gehörtraining: Tonhöhenvergleich</h1>
        <div class="container settings">
            <h2>Einstellungen</h2>
            <div>
                <p>Wähle den Schwierigkeitsgrad:</p>
                <div class="difficulty-options">
                    <button class="difficulty-btn active" data-
difficulty="easy">Einfach</button>
                    <button class="difficulty-btn" data-
difficulty="medium">Fortgeschritten</button>
                    <button class="difficulty-btn" data-
difficulty="hard">Profi</button>
                </div>
                <div class="difficulty-description" id="difficulty-
description">
                    Einfach: Große Terz (maximal 4 Halbtöne Unterschied)
                </div>
            </div>
            <div id="audio-permission" class="audio-status">
                Bitte klicke hier um Audio zu aktivieren
            </div>
```

```
          <button id="start-test" class="start-btn">Prüfung starten</button>
      </div>
```

```
      <div id="test-area" class="container hidden">
          <button id="play-tones" class="play-tones-btn">Töne
abspielen</button>
          <div class="answer-btns">
              <button id="higher-btn" class="answer-btn higher-
btn">Höher</button>
              <button id="lower-btn" class="answer-btn lower-
btn">Tiefer</button>
          </div>
          <div id="retry-next" class="hidden">
              <button id="retry-btn" class="play-tones-btn">Nochmal
hören</button>
              <button id="next-btn" class="next-btn">Nächste Frage</button>
          </div>
          <div class="progress">
              <div class="progress-dots" id="progress-dots"></div>
          </div>
          <div class="stats" id="stats">
              Frage 1 von 10 | Richtig: 0% | Falsch: 0%
          </div>
          <button id="cancel-test" class="cancel-btn">Prüfung
abbrechen</button>
      </div>
      <div id="final-result" class="container hidden">
          <h2>Prüfungsergebnis</h2>
          <div class="result-message" id="result-message"></div>
          <button id="restart-btn" class="start-btn">Neue Prüfung
starten</button>
      </div>
  </div>
```

```
  <script>
      // Warten, bis die Seite vollständig geladen ist
      window.addEventListener('DOMContentLoaded', () => {
          // DOM-Elemente
          const startTestBtn = document.getElementById('start-test');
          const testArea = document.getElementById('test-area');
          const playTonesBtn = document.getElementById('play-tones');
          const higherBtn = document.getElementById('higher-btn');
          const lowerBtn = document.getElementById('lower-btn');
          const progressDots = document.getElementById('progress-dots');
          const statsDiv = document.getElementById('stats');
          const finalResult = document.getElementById('final-result');
          const resultMessage = document.getElementById('result-message');
          const restartBtn = document.getElementById('restart-btn');
          const audioPermission = document.getElementById('audio-
permission');
          const cancelTestBtn = document.getElementById('cancel-test');
```

```
          const difficultyDescription = document.getElementById('difficulty-
description');
          const retryNextDiv = document.getElementById('retry-next');
          const retryBtn = document.getElementById('retry-btn');
          const nextBtn = document.getElementById('next-btn');
          // Audio Context
          let audioContext = null;
          let audioInitialized = false;
          // Schwierigkeitsgrad-Buttons
          const difficultyBtns = document.querySelectorAll('.ear-training-
app .difficulty-btn');
          // Variablen für das Spiel
          let difficulty = 'easy';  // Standardschwierigkeit: einfach
          let currentQuestion = 0;
          let correctAnswers = 0;
          let totalQuestions = 10;
          let questions = [];
          let currentFirstNote = null;
          let currentSecondNote = null;
          let isSecondNoteHigher = false;
          // Tonhöhen (C4 bis A5)
          const notes = ['C4', 'C#4', 'D4', 'D#4', 'E4', 'F4', 'F#4', 'G4',
'G#4', 'A4', 'A#4', 'B4',
                         'C5', 'C#5', 'D5', 'D#5', 'E5', 'F5', 'F#5', 'G5',
'G#5', 'A5'];
          // Frequenzen der Noten
          const noteFrequencies = {
              'C4': 261.63, 'C#4': 277.18, 'D4': 293.66, 'D#4': 311.13,
'E4': 329.63,
              'F4': 349.23, 'F#4': 369.99, 'G4': 392.00, 'G#4': 415.30,
'A4': 440.00,
              'A#4': 466.16, 'B4': 493.88, 'C5': 523.25, 'C#5': 554.37,
'D5': 587.33,
              'D#5': 622.25, 'E5': 659.25, 'F5': 698.46, 'F#5': 739.99,
'G5': 783.99,
              'G#5': 830.61, 'A5': 880.00
          };
          // Schwierigkeitsgradbeschreibungen
          const difficultyDescriptions = {
              'easy': 'Einfach: Große Terz (maximal 4 Halbtöne
Unterschied)',
              'medium': 'Fortgeschritten: Quinte (maximal 7 Halbtöne
Unterschied)',
              'hard': 'Profi: Oktave (maximal 12 Halbtöne Unterschied)'
          };
          // Audio initialisieren
          function initializeAudio() {
              if (!audioInitialized) {
                  try {
                      audioContext = new (window.AudioContext ||
window.webkitAudioContext)();
```

```
                    audioInitialized = true;
                    audioPermission.textContent = "Audio aktiviert ✓";
                    audioPermission.style.backgroundColor = "#DCEDC8";
                } catch (e) {
                    console.error("Web Audio API wird nicht unterstützt.",
e);
                    audioPermission.textContent = "Audio nicht verfügbar.
Bitte verwende einen modernen Browser.";
                    audioPermission.style.backgroundColor = "#FFCDD2";
                }
            }
        }
        // Audio-Erlaubnis anfordern
        audioPermission.addEventListener('click', initializeAudio);
        // Ton abspielen mit Web Audio API
        function playNote(note, delay = 0) {
            if (!audioContext) {
                alert("Bitte aktiviere zuerst Audio durch Klicken auf den
Audio-Status-Bereich.");
                return;
            }
            const frequency = noteFrequencies[note];
            if (!frequency) return;
            // Zeit für den Start des Tons
            const startTime = audioContext.currentTime + delay;
            // Oszillator erstellen
            const oscillator = audioContext.createOscillator();
            oscillator.type = 'sine'; // Sinuswelle für einen weicheren
Ton
            oscillator.frequency.setValueAtTime(frequency, startTime);
            // Envelope für einen Piano-ähnlichen Klang
            const gainNode = audioContext.createGain();
            gainNode.gain.setValueAtTime(0, startTime);
            gainNode.gain.linearRampToValueAtTime(0.7, startTime + 0.01);
            gainNode.gain.linearRampToValueAtTime(0.6, startTime + 0.1);
            gainNode.gain.exponentialRampToValueAtTime(0.001, startTime +
1.5);
            // Verbindungen herstellen
            oscillator.connect(gainNode);
            gainNode.connect(audioContext.destination);
            // Ton starten und stoppen
            oscillator.start(startTime);
            oscillator.stop(startTime + 1.5);
        }
        // Zwei Töne hintereinander abspielen
        function playTones() {
            if (!currentFirstNote || !currentSecondNote) return;
            // Audio initialisieren, falls noch nicht geschehen
            if (!audioInitialized) {
                initializeAudio();
            }
```

```
            // Alle Schaltflächen deaktivieren während des Abspielens
            higherBtn.disabled = true;
            lowerBtn.disabled = true;
            // Ersten Ton abspielen
            playNote(currentFirstNote, 0);
            // Zweiten Ton nach 1,5 Sekunden abspielen
            playNote(currentSecondNote, 1.5);
            // Nach 3 Sekunden Schaltflächen wieder aktivieren
            setTimeout(() => {
                higherBtn.disabled = false;
                lowerBtn.disabled = false;
            }, 3000);
        }
        // Zufälligen Ton zwischen C4 und A5 generieren
        function getRandomNote() {
            const randomIndex = Math.floor(Math.random() * notes.length);
            return notes[randomIndex];
        }
        // Zweiten Ton basierend auf dem Schwierigkeitsgrad generieren
        function generateSecondNote(firstNote) {
            const firstNoteIndex = notes.indexOf(firstNote);
            let maxDistance;
            // Maximale Entfernung basierend auf dem Schwierigkeitsgrad
            switch (difficulty) {
                case 'easy':     // Große Terz (4 Halbtöne)
                    maxDistance = 4;
                    break;
                case 'medium':   // Quinte (7 Halbtöne)
                    maxDistance = 7;
                    break;
                case 'hard':     // Oktave (12 Halbtöne)
                    maxDistance = 12;
                    break;
                default:
                    maxDistance = 4;
            }
            // Zufällige Richtung (höher oder tiefer)
            const direction = Math.random() < 0.5 ? -1 : 1;
            // Zufällige Distanz innerhalb der maximalen Entfernung
            let distance = Math.floor(Math.random() * maxDistance) + 1;
            distance *= direction;
            // Sicherstellen, dass der neue Index innerhalb des gültigen
Bereichs liegt
            let newIndex = firstNoteIndex + distance;
            if (newIndex < 0) newIndex = 0;
            if (newIndex >= notes.length) newIndex = notes.length - 1;
            // Vermeiden, dass beide Töne gleich sind
            if (newIndex === firstNoteIndex) {
                newIndex += (direction > 0) ? 1 : -1;
                if (newIndex < 0) newIndex = 1;
                if (newIndex >= notes.length) newIndex = notes.length - 2;
```

```
            }
            // Ist der zweite Ton höher?
            isSecondNoteHigher = newIndex > firstNoteIndex;
            return notes[newIndex];
        }
        // Neue Frage generieren
        function generateQuestion() {
            const firstNote = getRandomNote();
            const secondNote = generateSecondNote(firstNote);
            return {
                firstNote,
                secondNote,
                isSecondNoteHigher
            };
        }
        // Fortschrittsanzeige aktualisieren
        function updateProgress() {
            // Fortschrittspunkte aktualisieren
            progressDots.innerHTML = '';
            for (let i = 0; i < totalQuestions; i++) {
                const dot = document.createElement('div');
                dot.classList.add('dot');
                // Status des Punktes basierend auf dem Spielfortschritt
                if (i < currentQuestion) {
                    if (questions[i].correct) {
                        dot.classList.add('correct');
                    } else {
                        dot.classList.add('incorrect');
                    }
                } else if (i === currentQuestion) {
                    dot.classList.add('current');
                }
                progressDots.appendChild(dot);
            }
            // Statistiken aktualisieren
            const answered = Math.min(currentQuestion, totalQuestions);
            let correctPercentage = 0;
            let incorrectPercentage = 0;
            if (answered > 0) {
                correctPercentage = Math.round((correctAnswers / answered)
* 100);
                incorrectPercentage = 100 - correctPercentage;
            }
            statsDiv.textContent = `Frage ${currentQuestion + 1} von
${totalQuestions} | ` +
                                   `Richtig: ${correctPercentage}% | ` +
                                   `Falsch: ${incorrectPercentage}%`;
        }
        // Prüfung starten
        function startTest() {
            // Audio initialisieren, falls noch nicht geschehen
```

```
            if (!audioInitialized) {
                initializeAudio();
            }
            // Fragen generieren
            questions = [];
            for (let i = 0; i < totalQuestions; i++) {
                questions.push(generateQuestion());
            }
            // Variablen zurücksetzen
            currentQuestion = 0;
            correctAnswers = 0;
            // Aktuelle Frage setzen
            currentFirstNote = questions[0].firstNote;
            currentSecondNote = questions[0].secondNote;
            isSecondNoteHigher = questions[0].isSecondNoteHigher;
            // UI aktualisieren
            testArea.classList.remove('hidden');
            document.querySelector('.ear-training-app
.settings').classList.add('hidden');
            finalResult.classList.add('hidden');
            retryNextDiv.classList.add('hidden');
            higherBtn.classList.remove('correct', 'incorrect');
            lowerBtn.classList.remove('correct', 'incorrect');
            // Fortschritt aktualisieren
            updateProgress();
            // Töne automatisch abspielen
            setTimeout(playTones, 500);
        }
        // Prüfung abbrechen
        function cancelTest() {
            testArea.classList.add('hidden');
            document.querySelector('.ear-training-app
.settings').classList.remove('hidden');
        }
        // Zur nächsten Frage gehen
        function nextQuestion() {
            currentQuestion++;
            // Prüfen, ob alle Fragen beantwortet wurden
            if (currentQuestion >= totalQuestions) {
                showFinalResult();
                return;
            }
            // Aktuelle Frage setzen
            currentFirstNote = questions[currentQuestion].firstNote;
            currentSecondNote = questions[currentQuestion].secondNote;
            isSecondNoteHigher =
questions[currentQuestion].isSecondNoteHigher;
            // UI zurücksetzen
            retryNextDiv.classList.add('hidden');
            higherBtn.classList.remove('correct', 'incorrect');
            lowerBtn.classList.remove('correct', 'incorrect');
```

```
            // Fortschritt aktualisieren
            updateProgress();
            // Töne automatisch abspielen
            setTimeout(playTones, 500);
        }
        // Endergebnis anzeigen
        function showFinalResult() {
            testArea.classList.add('hidden');
            finalResult.classList.remove('hidden');
            const percentage = Math.round((correctAnswers /
totalQuestions) * 100);
            let message;
            if (percentage >= 90) {
                message = `Ausgezeichnet! Du hast ${percentage}% der
Fragen richtig beantwortet!`;
            } else if (percentage >= 70) {
                message = `Gut gemacht! Du hast ${percentage}% der Fragen
richtig beantwortet.`;
            } else if (percentage >= 50) {
                message = `Nicht schlecht! Du hast ${percentage}% der
Fragen richtig beantwortet.`;
            } else {
                message = `Du hast ${percentage}% der Fragen richtig
beantwortet. Übe weiter!`;
            }
            resultMessage.textContent = message;
        }
        // Schwierigkeitsgradbeschreibung aktualisieren
        function updateDifficultyDescription(selectedDifficulty) {
            difficultyDescription.textContent =
difficultyDescriptions[selectedDifficulty];
        }
        // Event-Listener
        // Schwierigkeitsgrad wählen
        difficultyBtns.forEach(btn => {
            btn.addEventListener('click', () => {
                // Aktiven Button zurücksetzen
                difficultyBtns.forEach(b => b.classList.remove('active'));
                // Neuen Button aktivieren
                btn.classList.add('active');
                // Schwierigkeitsgrad setzen
                difficulty = btn.dataset.difficulty;
                // Beschreibung aktualisieren
                updateDifficultyDescription(difficulty);
            });
        });
        // Prüfung starten
        startTestBtn.addEventListener('click', startTest);
        // Töne abspielen
        playTonesBtn.addEventListener('click', playTones);
        // Prüfung abbrechen
```

```
        cancelTestBtn.addEventListener('click', cancelTest);
        // Antwort: Höher
        higherBtn.addEventListener('click', () => {
            // Antwort überprüfen
            const isCorrect = isSecondNoteHigher;
            // Antwort speichern
            questions[currentQuestion].correct = isCorrect;
            if (isCorrect) {
                correctAnswers++;
                higherBtn.classList.add('correct');
                // Bei richtiger Antwort nach 1 Sekunde automatisch zur
nächsten Frage
                setTimeout(nextQuestion, 1000);
            } else {
                higherBtn.classList.add('incorrect');
                // Bei falscher Antwort Optionen anzeigen
                retryNextDiv.classList.remove('hidden');
            }
            // Fortschritt aktualisieren
            updateProgress();
        });
        // Antwort: Tiefer
        lowerBtn.addEventListener('click', () => {
            // Antwort überprüfen
            const isCorrect = !isSecondNoteHigher;
            // Antwort speichern
            questions[currentQuestion].correct = isCorrect;
            if (isCorrect) {
                correctAnswers++;
                lowerBtn.classList.add('correct');
                // Bei richtiger Antwort nach 1 Sekunde automatisch zur
nächsten Frage
                setTimeout(nextQuestion, 1000);
            } else {
                lowerBtn.classList.add('incorrect');
                // Bei falscher Antwort Optionen anzeigen
                retryNextDiv.classList.remove('hidden');
            }
            // Fortschritt aktualisieren
            updateProgress();
        });
        // "Nochmal hören" Button
        retryBtn.addEventListener('click', () => {
            // Die gleichen Töne erneut abspielen
            playTones();
        });
        // "Nächste Frage" Button
        nextBtn.addEventListener('click', () => {
            // Zur nächsten Frage gehen
            nextQuestion();
        });
```

```
          // Prüfung neu starten
          restartBtn.addEventListener('click', () => {
              finalResult.classList.add('hidden');
              document.querySelector('.ear-training-app
.settings').classList.remove('hidden');
          });
      });
  </script>
```

</body> </html>