Blended-Learning-Masterstudiengang
Programmierung von Videospielen

tech technologische universität

tech technologische universität

Blended-Learning-Masterstudiengang

Programmierung von Videospielen

Modalität: **Blended Learning (Online + Praktikum)**
Dauer: **12 Monate**
Qualifizierung: **TECH Global University**
Kreditpunkte: **60 + 4 ECTS**
Internetzugang: www.techtitute.com/DE/videospiele/semiprasentieller-masterstudiengang/semiprasentieller-masterstudiengang-programmierung-videospielen

Index

01

Präsentation

Der Sektor der Videospielentwicklung ist mehr als konsolidiert und zieht immer mehr junge Menschen an. Eine der Säulen, auf die sich ein erfolgreiches Videospiel stützt, ist die Programmierung. Die Verarbeitung, die die grundlegenden Anweisungen erstellt und die allgemeine Funktionsweise diktiert, ist für das perfekte Funktionieren der Geschichte und der Entwicklung des Spiels unerlässlich. Dieser Blended-Learning-Studiengang bietet Spieleprofis eine Spezialisierung, die es ihnen ermöglicht, ihre Karriere in der *Gaming*-Branche voranzutreiben. Es handelt sich um ein 100%iges Online-Studium, das an ihre Bedürfnisse angepasst ist und ein Praktikum in einem Studio für die Kreation und Entwicklung von Videospielen beinhaltet. All dies ermöglicht es ihnen, in ihrer beruflichen Laufbahn mit den besten Experten voranzukommen.

Werden Sie mit diesem Blended-Learning-Masterstudiengang zu einem der besten Programmierer in der Videospielbranche“

Die Videospielindustrie hat ein großes Potenzial. Die steigende Nachfrage und die Ansprüche der *Gamer* selbst haben in dieser Branche zu einem Wettlauf um die Perfektion ihrer Spiele geführt. Das hohe Qualitätsniveau jeder einzelnen Kreation wird durch ein Team von Programmierern mit hervorragenden Qualifikationen unterstützt.

Dieser Blended-Learning-Masterstudiengang in Programmierung von Videospielen ist eine Antwort auf die aktuellen Bedürfnisse des Marktes, der zunehmend spezialisierte Fachleute mit einem hohen Grad an Beteiligung an den Kreationen verlangt. Kreativität spielt eine wichtige Rolle, aber ohne solides Wissen ist es nicht möglich, Videospiele auf hohem Niveau zu entwickeln.

Aus diesem Grund bietet dieser Studiengang den Studenten ein umfassendes Wissen über die Grundlagen der Programmierung und des Software-*Engineerings*, befasst sich mit der Struktur von Daten und Algorithmen und lehrt objektorientierte Programmierung und *Engine*-Spezifikationen. Dieser Studiengang befasst sich auch mit der Echtzeitprogrammierung, um dem Videospielprofi einen vollständigen Blended-Learning-Masterstudiengang zu bieten.

Um das Ziel zu erreichen, in der beruflichen Laufbahn der Videospielprogrammierung voranzukommen, werden die Studenten von einem fachkundigen Lehrkörper in diesem Bereich betreut, der sie jederzeit anleitet und unterrichtet. Darüber hinaus ergänzen die interaktiven Inhalte mit Videozusammenfassungen, Fallstudien und zusätzlicher Lektüre den umfangreichen Lehrplan, den TECH in diesem 100%igen Online-Studium mit Praktika in Unternehmen zur Verfügung stellt.

So können die Studenten diesen Studiengang mit einem 3-wöchigen Intensivaufenthalt in einem führenden Studio für Videospielprogrammierung abschließen. Ein ideales berufliches Umfeld, in dem sie die ausgefeiltesten Arbeitsmethoden, Programme und Techniken für die Erstellung hochwertiger Titel vor Ort testen können. Eine einzigartige Gelegenheit, die Ihnen nur TECH, die größte digitale Universität der Welt, bieten kann.

Dieser **Blended-Learning-Masterstudiengang in Programmierung von Videospielen** enthält das vollständigste und aktuellste Programm auf dem Markt. Seine herausragendsten Merkmale sind:

- Entwicklung von mehr als 100 Fallstudien zur Videospielprogrammierung, die von Programmierern und Universitätsprofessoren mit umfassender Erfahrung in der Videospielbranche präsentiert werden
- Sein anschaulicher, schematischer und äußerst praktischer Inhalt soll wissenschaftliche Informationen zu den Disziplinen liefern, die für die berufliche Praxis unerlässlich sind
- Die Entwicklung von Fallstudien, die von Experten für Programmierung und Entwicklung von Videospielen präsentiert werden
- Der anschauliche, schematische und äußerst praxisnahe Inhalt vermittelt alle für die berufliche Praxis unverzichtbaren wissenschaftlichen und praktischen Informationen
- Praktische Übungen, bei denen der Selbstbewertungsprozess zur Verbesserung des Lernens genutzt werden kann
- Sein besonderer Schwerpunkt liegt auf innovativen Methoden
- Theoretische Lektionen, Fragen an den Experten, Diskussionsforen zu kontroversen Themen und individuelle Reflexionsarbeit
- Die Verfügbarkeit des Zugangs zu Inhalten von jedem festen oder tragbaren Gerät mit Internetanschluss
- Ergänzt wird dies durch theoretische Vorträge, Fragen an den Experten, Diskussionsforen zu kontroversen Themen und individuelle Reflexionsarbeit
- Verfügbarkeit der Inhalte von jedem festen oder tragbaren Gerät mit einer Internetverbindung
- Außerdem haben Sie die Möglichkeit, ein klinisches Praktikum in einem der besten Kreativstudios zu absolvieren

Dieses 100%ige Online-Programm bietet Ihnen die Möglichkeit, ein Praktikum in einem Studio zu absolvieren und sich mit den besten Programmierern zu messen“

Dieser Blended-Learning-Masterstudiengang ermöglicht es Ihnen, sich im Bereich der Videospielprogrammierung zu profilieren. Schreiben Sie sich jetzt ein.

Entwickeln Sie mit diesem Blended-Learning-Masterstudiengang effiziente Anwendungen für Videospiel-Engines.

Dieser Masterstudiengang mit berufsbezogenem Charakter und Blended-Learning-Modalität zielt auf die Aktualisierung von Videospielprofis ab, die in großen Kreativstudios arbeiten und ein hohes Qualifikationsniveau benötigen. Die Inhalte basieren auf den neuesten wissenschaftlichen Erkenntnissen und sind didaktisch darauf ausgerichtet, theoretisches Wissen in die Praxis der *Gaming*-Branche zu integrieren. Die theoretisch-praktischen Elemente erleichtern die Aktualisierung des Wissens und ermöglichen die Entscheidungsfindung in der Videospielprogrammierung.

Dank seiner multimedialen Inhalte, die mit der neuesten Bildungstechnologie entwickelt wurden, wird der Videospielfachkraft ein situiertes und kontextbezogenes Lernen ermöglicht, d. h. eine simulierte Umgebung, die eine immersive Fortbildung bietet, die auf die Ausführung von realen Situationen ausgerichtet ist. Das Konzept dieses Studiengangs konzentriert sich auf problemorientiertes Lernen, bei dem sie versuchen muss, die verschiedenen Situationen aus der beruflichen Praxis zu lösen, die während des Studiengangs auftreten. Zu diesem Zweck wird sie von einem innovativen interaktiven Videosystem unterstützt, das von renommierten Experten entwickelt wurde.

02

Warum dieses Programm belegen?

Die Programmierung erfordert solide theoretische Kenntnisse, aber zweifellos ist es die tägliche Praxis, die die Beherrschung des Videospielsektors perfektioniert. Aus diesem Grund hat TECH diesen Studiengang geschaffen, der den fortschrittlichsten Online-Lehrplan in Bereichen wie Algorithmusdesign, intelligente Systeme und Echtzeitprogrammierung perfekt mit einem praktischen Aufenthalt in einem führenden Programmierstudio kombiniert. Auf diese Weise erhalten die Studenten einen viel umfassenderen Einblick in die Videospielprogrammierung, die verwendeten Programme und die neuesten Techniken. All dies unter der ständigen Anleitung der besten Experten auf diesem Gebiet.

TECH bietet Ihnen die Möglichkeit, ein effektives und intensives Praktikum in der Programmierung von Videospielen zu absolvieren“

1. Aktualisierung basierend auf der neuesten verfügbaren Technologie

In den letzten Jahren hat die Technologie den Bereich der Videospielprogrammierung revolutioniert und die Kreation von Spielen mit höherer Qualität und Realismus begünstigt. Aus diesem Grund und um den Studenten diese Technologie näher zu bringen, hat TECH diesen Blended-Learning-Masterstudiengang geschaffen, in dem die Fachleute etwas über Videospiel-Engines, aktuelle technologische Herausforderungen und Software für die Erstellung von Ontologien lernen werden. Auf diese Weise werden sie auf den neuesten Stand der verfügbaren Technologie gebracht.

2. Auf die Erfahrung der besten Spezialisten zurückgreifen

Im Rahmen dieses Studiengangs werden die Studenten stets von den besten Spezialisten der Videospielprogrammierung betreut. So werden sie während der theoretischen Phase von einem exzellenten Lehrteam mit umfassender Erfahrung in diesem Bereich begleitet, während sie während des Aufenthalts vor Ort von echten Experten betreut werden, die zum Team des Studios gehören, in dem sie die praktische Phase absolvieren werden.

3. Einstieg in erstklassige Umgebungen

TECH führt ein strenges Auswahlverfahren für alle Studios und Unternehmen durch, in denen die Praktika durchgeführt werden. Auf diese Weise wird den Studenten der Zugang zu einem professionellen Umfeld für die Programmierung von Videospielen auf höchstem Niveau garantiert. Auf diese Weise können sie aus erster Hand erfahren, wie der Arbeitsalltag eines spezialisierten Programmierers aussieht, und die Techniken und Methoden kennen lernen, die für die Entwicklung von hochwertigen Spielen verwendet werden.

4. Kombination der besten Theorie mit modernster Praxis

TECH passt sich mit diesem Studiengang an die tägliche Arbeit von Programmierern an und verfolgt daher einen theoretisch-praktischen Ansatz, der sich nicht auf lange Studienzeiten, sondern auf Schlüsselkonzepte konzentriert. Diese akademische Einrichtung bietet somit ein innovatives Lernmodell, das es den Studenten ermöglicht, sich die notwendigen Kenntnisse anzueignen, um bei der Programmierung hochwertiger Videospiele die Führung zu übernehmen.

5. Ausweitung der Grenzen des Wissens

TECH bietet die Möglichkeit, diese praktische Ausbildung nicht nur in Zentren von nationaler, sondern auch von internationaler Bedeutung zu absolvieren. Auf diese Weise kann der Spezialist seine Grenzen erweitern und mit den besten Fachleuten, die in erstklassigen Krankenhäusern auf verschiedenen Kontinenten praktizieren, gleichziehen. Eine einzigartige Gelegenheit, die nur TECH, die größte digitale Universität der Welt, bieten kann.

Sie werden in dem Zentrum Ihrer Wahl vollständig in die Praxis eintauchen"

03

Ziele

Die Gestaltung dieses Blended-Learning-Masterstudiengangs wird es den Studenten ermöglichen, die notwendigen Fähigkeiten zu erwerben, um in der hart umkämpften Programmierung, die qualifiziertes Personal erfordert, voranzukommen. Daher besteht das Hauptziel dieses Kurses darin, sicherzustellen, dass die Studenten hervorragende Videospielentwickler werden, indem sie alle Werkzeuge nutzen, die dieser Kurs bietet. Um dies zu erreichen, bietet dieser Studiengang aktualisierte Multimediainhalte mit zusätzlicher Lektüre und ein *Relearning*-Lernsystem, das auf der Wiederholung von Inhalten basiert und die Festigung der Konzepte erleichtern soll.

Möchten Sie ein Multiplayer-Spiel starten? Lernen Sie, wie Sie Ihre Idee in eine echte Programmiersprache für diese Art von Videospiel umsetzen können“

Allgemeines Ziel

- Die Studenten sollen mit den verschiedenen Programmiersprachen und -methoden vertraut gemacht werden, die für Videospiele verwendet werden. Zu diesem Zweck wird der Produktionsprozess eines Spiels und seine Integration in die verschiedenen Phasen eingehend untersucht. Ebenso wird der Videospielprofi die Grundlagen des Videospiel-Designs und die wichtigsten theoretischen Kenntnisse erlernen. Darüber hinaus wird er am Ende dieses Kurses in der Lage sein, die Rolle der Programmierung zu verstehen und Web- und Multiplayer-Videospiele zu entwickeln

Dieses Programm ermöglicht Ihnen einen beruflichen Aufstieg. Mit diesem Blended-Learning-Masterstudiengang werden Sie in der Lage sein, alle Datenstrukturen und Algorithmen zu beherrschen"

Spezifische Ziele

Modul 1. Grundlagen der Programmierung

- Verstehen der grundlegenden Struktur eines Computers, von Software und allgemeinen Programmiersprachen
- Analysieren der wesentlichen Elemente eines Computerprogramms, wie z. B. die verschiedenen Datentypen, Operatoren, Ausdrücke, Anweisungen, E/A und Steueranweisungen
- Interpretieren von Algorithmen, die die notwendige Grundlage für die Entwicklung von Computerprogrammen sind

Modul 2. Datenstruktur und Algorithmen

- Erlernen der wichtigsten Strategien für den Entwurf von Algorithmen sowie der verschiedenen Methoden und Maße für die Berechnung von Algorithmen
- Unterscheiden der Funktionsweise von Algorithmen, ihrer Strategie und Beispiele für ihren Einsatz bei den wichtigsten bekannten Problemen
- Verstehen der *Backtracking*-Technik und ihrer wichtigsten Anwendungen

Modul 3. Objektorientierte Programmierung

- Kennen der verschiedenen Entwurfsmuster für objektorientierte Probleme
- Verstehen der Bedeutung von Dokumentation und Tests bei der Softwareentwicklung
- Beherrschen der Verwendung von Threads und Synchronisation sowie die Lösung gängiger Probleme bei der nebenläufigen Programmierung

Modul 4. Videospielkonsolen und -geräte

- Kennen der grundlegenden Funktionsweise der wichtigsten Ein- und Ausgabeperipheriegeräte
- Verstehen der wichtigsten Auswirkungen der verschiedenen Plattformen auf das Design
- Untersuchen des Aufbaus, der Organisation, des Betriebs und der Verbindung von Geräten und Systemen
- Verstehen der Rolle des Betriebssystems und der Entwicklungskits für mobile Geräte und Videospielplattformen

Modul 5. Softwareentwicklung

- Kennen der Grundlagen der Softwareentwicklung sowie des Softwareprozesses und der verschiedenen Modelle für dessen Entwicklung, einschließlich agiler Technologien
- Kennen des *Requirements Engineerings*, seiner Entwicklung, Ausarbeitung, Verhandlung und Validierung, um die wichtigsten Standards in Bezug auf Softwarequalität und Projektmanagement verstehen

Modul 6. Videospiel-Engines

- Entdecken der Funktionsweise und Architektur einer Videospiel-Engine
- Verstehen der grundlegenden Eigenschaften bestehender Spiel-Engines
- Richtiges und effizientes Programmieren von Anwendungen für Videospiel-Engines
- Auswählen des am besten geeigneten Paradigmas und der Programmiersprachen für die Programmierung von Anwendungen, die auf Videospiel-Engines angewendet werden

Modul 7. Intelligente Systeme

- Kennen aller Konzepte im Zusammenhang mit der Agententheorie und der Agentenarchitektur sowie deren Argumentationsprozess
- Verstehen der Theorie und Praxis, die hinter den Konzepten von Information und Wissen stehen, sowie der verschiedenen Arten der Darstellung von Wissen
- Verstehen, wie semantische Reasoner, wissensbasierte Systeme und Expertensysteme funktionieren

Modul 8. Programmierung in Echtzeit

- Analysieren der wichtigsten Merkmale einer Echtzeit-Programmiersprache, die sie von einer herkömmlichen Programmiersprache unterscheiden
- Verstehen der grundlegenden Konzepte von Computersystemen
- Erlangen der Fähigkeit, die wichtigsten Grundlagen und Techniken der Echtzeitprogrammierung anzuwenden

Modul 9. Design und Entwicklung von Webspielen

- In der Lage sein, Spiele und interaktive Webanwendungen mit der entsprechenden Dokumentation zu entwerfen
- Bewerten der wichtigsten Merkmale von Spielen und interaktiven Webanwendungen, um professionell und korrekt zu kommunizieren

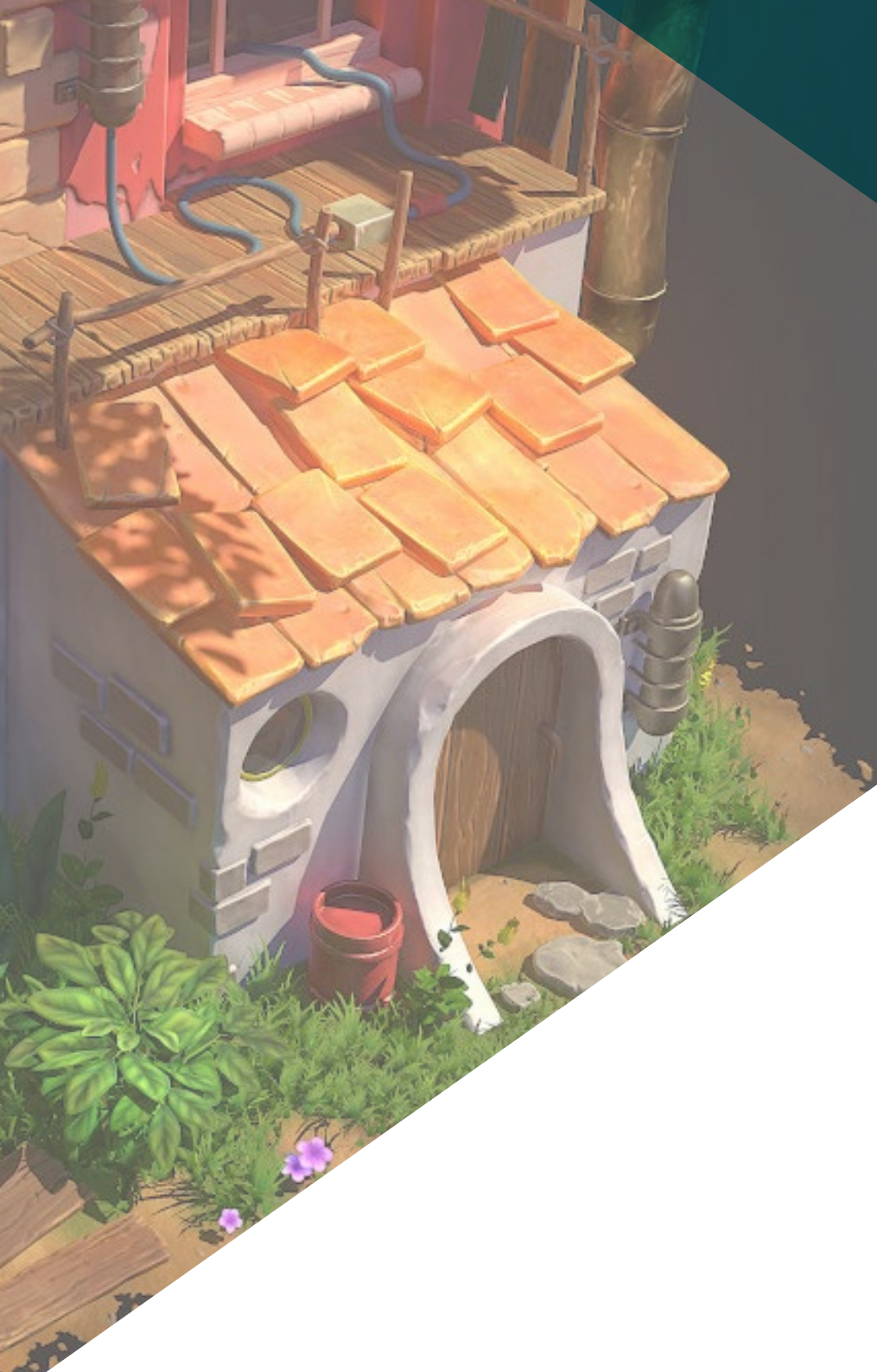
Modul 10. Multiplayer-Netzwerke und -Systeme

- Beschreiben der Architektur des Transmission Control Protocol/Internet Protocol (TCP/IP) und die grundlegende Funktionsweise von drahtlosen Netzwerken
- Analysieren der Sicherheit in Bezug auf Videospiele
- Erwerben der Fähigkeit, Multiplayer-Online-Spiele zu entwickeln

04

Kompetenzen

Der Videospielprofi, der diesen Blended-Learning-Masterstudiengang absolviert, erwirbt eine Reihe von Fähigkeiten, die ihm den Einstieg in jede Art von Videospielentwicklungsstudio ermöglichen. So werden die Studenten am Ende dieses Kurses über die notwendigen Kompetenzen verfügen, um in verschiedenen Sprachen zu programmieren, die in der *Gaming*-Industrie verwendet werden, und sie werden die wesentlichen Fähigkeiten erwerben, um ein Projekt von Anfang bis Ende auf verschiedenen Plattformen und Videospiel-Engines durchzuführen.

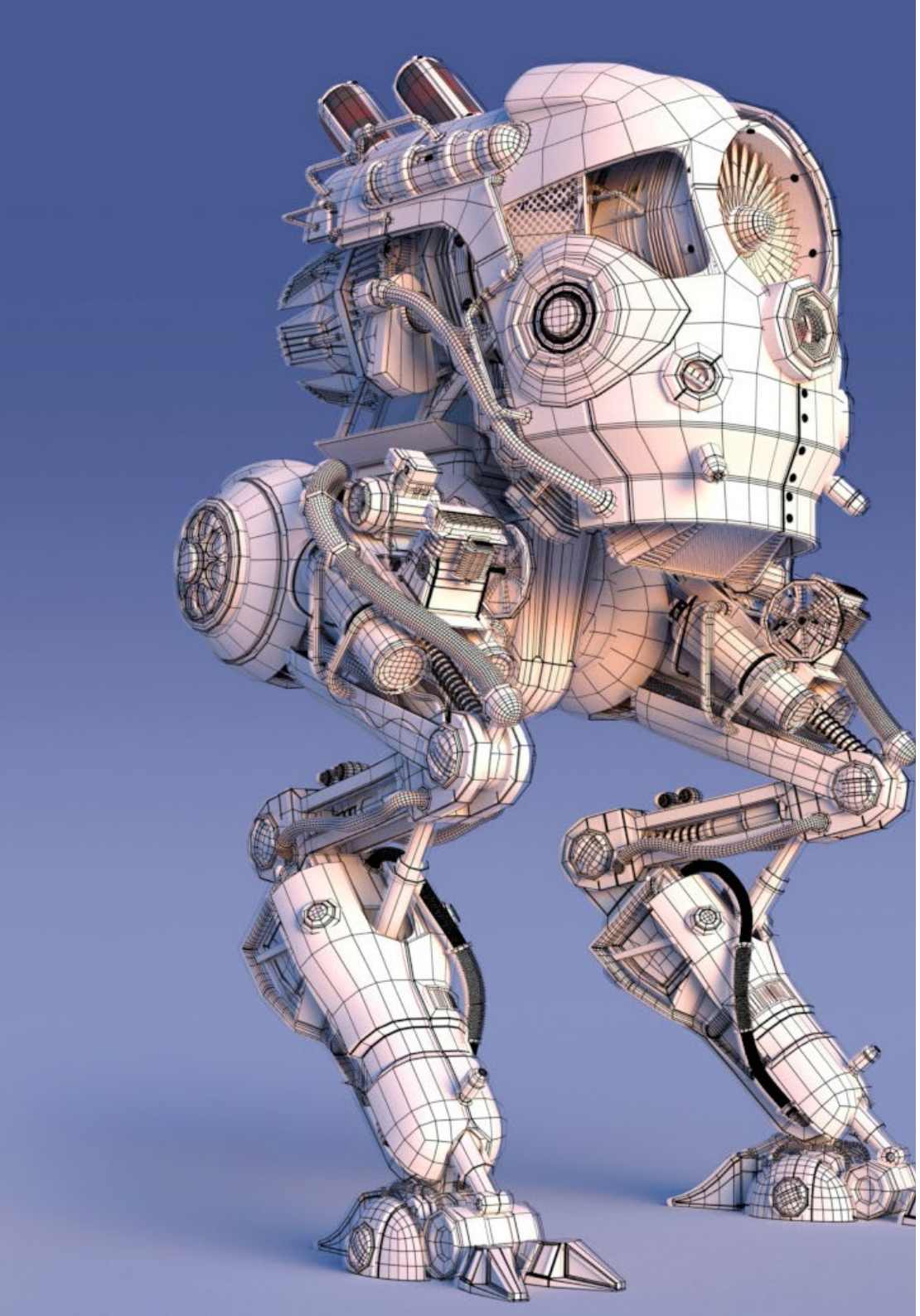


Erwerben Sie die notwendigen Fähigkeiten, um Videospiele für Xbox One, PS4 und Wii U-Switch zu programmieren"

Allgemeine Kompetenzen

- Entwerfen aller Phasen eines Videospiels, von der ersten Idee bis zur endgültigen Veröffentlichung
- Spezialisierung als Videospielprogrammierer
- Vertiefen in alle Teile der Entwicklung, von der anfänglichen Architektur über die Programmierung des Spielercharakters bis hin zu allen am Spielprozess beteiligten Elementen
- Erwerben einer Gesamtvision des Projekts und in der Lage zu sein, Lösungen für die verschiedenen Probleme und Herausforderungen zu finden, die bei der Entwicklung eines Videospiels auftreten

Bevor Sie programmieren, sollten Sie sich mit dem Spielerlebnis vertraut machen und die Spielbarkeit des Videospiels analysieren. Mit diesem Blended-Learning-Masterstudiengang lernen Sie, wie Sie erfolgreich sein können“



Spezifische Kompetenzen

- Kennen der Software, die notwendig ist, um ein Profi in der Entwicklung von Videospielen zu sein
- Verstehen der Erfahrung des Spielers und wissen, wie man den Spielverlauf analysiert
- Verstehen aller theoretischen und praktischen Vorgehensweisen bei der Programmierung von Videospielen
- Beherrschen der nützlichsten Programmiersprachen für die Welt der Videospiele
- Integrieren der erlernten Programmierung in verschiedene Arten von Konsolen und Plattformen
- Programmieren von Web- und Multiplayer-Videospielen
- Verinnerlichen des Konzepts der Videospiel-Engine, um korrekt programmieren zu können
- Anwenden von Kenntnissen der Softwaretechnik auf die Programmierung von Videospielen

05

Struktur und Inhalt

Der Blended-Learning-Masterstudiengang in Programmierung von Videospielen bietet den Studenten eine umfassende und breit gefächerte inhaltliche Fortbildung in der Videospielentwicklung, die in 10 spezialisierten Modulen strukturiert ist. Die Lehrkräfte dieses Studiengangs können die Grundlagen der Programmierung vertiefen und gleichzeitig aktuelle, praxisorientierte Kenntnisse erwerben. Auf diese Weise erhalten die Studenten einen qualitativ hochwertigen Unterricht mit innovativen Inhalten in multimedialem Format, die jederzeit heruntergeladen und konsultiert werden können. Die theoretische Fortbildung, die zu 100% online stattfindet, gibt Ihnen die Freiheit, das Lehrpensum nach Ihren Bedürfnissen zu verteilen.

Ein Lehrplan, der Ihnen die fortschrittlichsten und aktuellsten Inhalte im Bereich der Videospielprogrammierung bietet“

Modul 1. Grundlagen der Programmierung

1.1. Einführung in die Programmierung
- 1.1.1. Grundlegende Struktur eines Computers
- 1.1.2. Software
- 1.1.3. Programmiersprachen
- 1.1.4. Lebenszyklus einer Softwareanwendung

1.2. Algorithmusentwurf
- 1.2.1. Lösung von Problemen
- 1.2.2. Deskriptive Techniken
- 1.2.3. Elemente und Struktur eines Algorithmus

1.3. Elemente eines Programms
- 1.3.1. Ursprung und Merkmale der Sprache C++
- 1.3.2. Die Entwicklungsumgebung
- 1.3.3. Konzept des Programms
- 1.3.4. Arten von grundlegender Daten
- 1.3.5. Betreiber
- 1.3.6. Ausdrücke
- 1.3.7. Sätze
- 1.3.8. Dateneingabe und -ausgabe

1.4. Kontrollsätze
- 1.4.1. Sätze
- 1.4.2. Verzweigungen
- 1.4.3. Schleifen

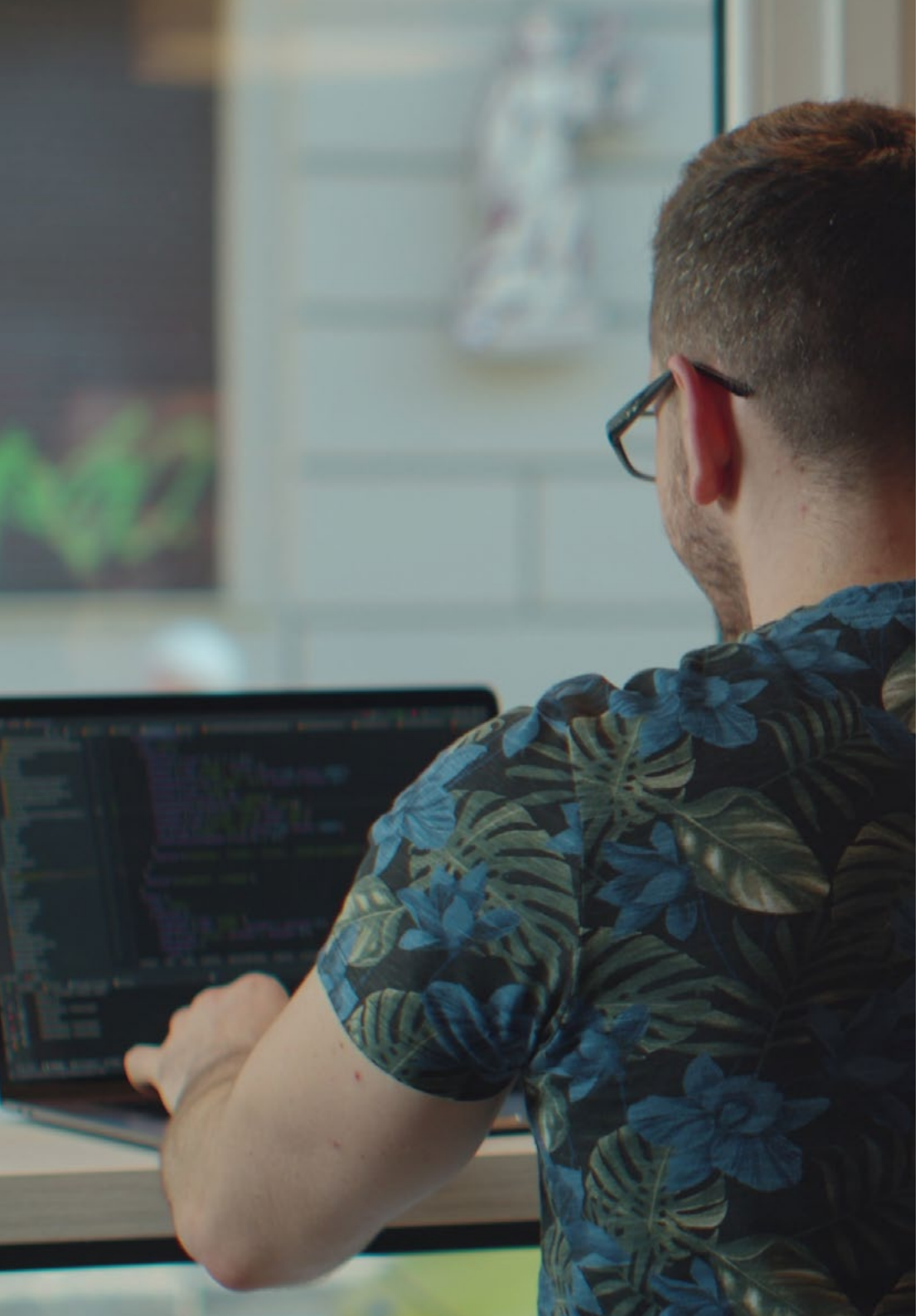
1.5. Abstraktion und Modularität: Funktionen
- 1.5.1. Modularer Aufbau
- 1.5.2. Konzept der Funktion und des Nutzens
- 1.5.3. Definition einer Funktion
- 1.5.4. Ausführungsablauf beim Aufruf einer Funktion
- 1.5.5. Prototyp einer Funktion
- 1.5.6. Rückgabe der Ergebnisse
- 1.5.7. Aufrufen einer Funktion: Parameter
- 1.5.8. Übergabe von Parametern per Referenz und per Wert
- 1.5.9. Kennung des Geltungsbereichs

1.6. Statische Datenstrukturen
- 1.6.1. *Arrays*
- 1.6.2. Matrizen. Polyeder
- 1.6.3. Suchen und Sortieren
- 1.6.4. Zeichenketten. E/A-Funktionen für Zeichenketten
- 1.6.5. Strukturen. Verbindungen
- 1.6.6. Neue Datentypen

1.7. Dynamische Datenstrukturen: Zeiger
- 1.7.1. Konzept. Definition von Zeiger
- 1.7.2. Operatoren und Operationen mit Zeigern
- 1.7.3. *Arrays* von Zeigern
- 1.7.4. Zeiger und *Arrays*
- 1.7.5. Zeiger auf Zeichenketten
- 1.7.6. Zeiger auf Strukturen
- 1.7.7. Multiple Indirektion
- 1.7.8. Zeiger auf Funktionen
- 1.7.9. Übergabe von Funktionen, Strukturen und *Arrays* als Funktionsparameter

1.8. Dateien
- 1.8.1. Grundlegende Konzepte
- 1.8.2. Dateioperationen
- 1.8.3. Datentypen
- 1.8.4. Organisation von Dateien
- 1.8.5. Einführung in C++ Dateien
- 1.8.6. Handhabung von Dateien

1.9. Rekursion
- 1.9.1. Definition von Rekursion
- 1.9.2. Arten der Rekursion
- 1.9.3. Vor- und Nachteile
- 1.9.4. Überlegungen
- 1.9.5. Rekursiv-iterative Umwandlung
- 1.9.6. Der Rekursionsstapel



1.10. Prüfung und Dokumentation
- 1.10.1. Programm-Tests
- 1.10.2. White-Box-Tests
- 1.10.3. Black-Box-Tests
- 1.10.4. Test-Tools
- 1.10.5. Programm-Dokumentation

Modul 2. Datenstruktur und Algorithmen

2.1. Einführung in Algorithmus-Design-Strategien
- 2.1.1. Rekursion
- 2.1.2. Aufteilen und erobern
- 2.1.3. Andere Strategien

2.2. Effizienz und Analyse von Algorithmen
- 2.2.1. Maßnahmen zur Steigerung der Effizienz
- 2.2.2. Messung der Eingabegröße
- 2.2.3. Messung der Ausführungszeit
- 2.2.4. Schlimmster, bester und durchschnittlicher Fall
- 2.2.5. Asymptotische Notation
- 2.2.6. Kriterien für die mathematische Analyse von nicht-rekursiven Algorithmen
- 2.2.7. Mathematische Analyse von rekursiven Algorithmen
- 2.2.8. Empirische Analyse von Algorithmen

2.3. Sortieralgorithmen
- 2.3.1. Konzept der Sortierung
- 2.3.2. Sortieren der Blase
- 2.3.3. Sortieren nach Auswahl
- 2.3.4. Reihenfolge der Insertion
- 2.3.5. Sortierung zusammenführen (Merge_Sort)
- 2.3.6. Schnelle Sortierung (Quick_Sort)

2.4. Algorithmen mit Bäumen
- 2.4.1. Konzept des Baumes
- 2.4.2. Binäre Bäume
- 2.4.3. Baumpfade
- 2.4.4. Ausdrücke darstellen
- 2.4.5. Geordnete binäre Bäume
- 2.4.6. Ausgeglichene binäre Bäume

2.5. Algorithmen mit *Heaps*
- 2.5.1. *Heaps*
- 2.5.2. Der *Heapsort*-Algorithmus
- 2.5.3. Prioritätswarteschlangen

2.6. Graph-Algorithmen
- 2.6.1. Vertretung
- 2.6.2. Lauf in Breite
- 2.6.3. Lauf in Tiefe
- 2.6.4. Topologische Anordnung

2.7. *Greedy*-Algorithmen
- 2.7.1. Die *Greedy*-Strategie
- 2.7.2. Elemente der *Greedy*-Strategie
- 2.7.3. Währungsumtausch
- 2.7.4. Das Problem des Reisenden
- 2.7.5. Problem mit dem Rucksack

2.8. Minimale Pfadsuche
- 2.8.1. Das Problem des minimalen Pfades
- 2.8.2. Negative Bögen und Zyklen
- 2.8.3. Dijkstra-Algorithmus

2.9. *Greedy*-Algorithmen auf Graphen
- 2.9.1. Der minimal aufspannende Baum
- 2.9.2. Algorithmus von Prim
- 2.9.3. Algorithmus von Kruskal
- 2.9.4. Komplexitätsanalyse

2.10. *Backtracking*
- 2.10.1. Das *Backtracking*
- 2.10.2. Alternative Techniken

Modul 3. Objektorientierte Programmierung

3.1. Einführung in die objektorientierte Programmierung
- 3.1.1. Einführung in die objektorientierte Programmierung
- 3.1.2. Klassen-Design
- 3.1.3. Einführung in UML für die Modellierung von Problemen

3.2. Beziehungen zwischen Klassen
- 3.2.1. Abstraktion und Vererbung
- 3.2.2. Fortgeschrittene Konzepte der Vererbung
- 3.2.3. Polymorphismen
- 3.2.4. Zusammensetzung und Aggregation

3.3. Einführung in Entwurfsmuster für objektorientierte Probleme
- 3.3.1. Was sind Entwurfsmuster?
- 3.3.2. *Factory*-Muster
- 3.3.3. *Singleton*-Muster
- 3.3.4. *Observer*-Muster
- 3.3.5. *Composite*-Muster

3.4. Ausnahmen
- 3.4.1. Was sind Ausnahmen?
- 3.4.2. Abfangen und Behandlung von Ausnahmen
- 3.4.3. Start von Ausnahmen
- 3.4.4. Erstellung von Ausnahmen

3.5. Benutzeroberflächen
- 3.5.1. Einführung in Qt
- 3.5.2. Positionierung
- 3.5.3. Was sind Ereignisse?
- 3.5.4. Ereignisse: Definition und Erfassung
- 3.5.5. Entwicklung von Benutzeroberflächen

3.6. Einführung in die gleichzeitige Programmierung
- 3.6.1. Einführung in die gleichzeitige Programmierung
- 3.6.2. Der Prozess und das *Thread*-Konzept
- 3.6.3. Interaktion zwischen Prozessen oder *Threads*
- 3.6.4. *Threads* in C++
- 3.6.5. Vor- und Nachteile der gleichzeitigen Programmierung

3.7. *Thread*-Verwaltung und Synchronisierung
- 3.7.1. Lebenszyklus eines *Threads*
- 3.7.2. Die Klasse *Thread*
- 3.7.3. Planung des *Threads*
- 3.7.4. Gruppen von *Threads*
- 3.7.5. Daemon-*Threads*
- 3.7.6. Synchronisierung
- 3.7.7. Verriegelungsmechanismen
- 3.7.8. Kommunikationsmechanismen
- 3.7.9. Monitore

3.8. Häufige Probleme bei der gleichzeitigen Programmierung
- 3.8.1. Das Erzeuger-Verbraucher-Problem
- 3.8.2. Das Problem von Lesern und Schriftstellern
- 3.8.3. Das Problem der speisenden Philosophen

3.9. Software-Dokumentation und -Tests
- 3.9.1. Warum ist es wichtig, Software zu dokumentieren?
- 3.9.2. Design-Dokumentation
- 3.9.3. Verwendung von Tools zur Dokumentation

3.10. Software-Tests
- 3.10.1. Einführung in das Testen von Software
- 3.10.2. Arten von Tests
- 3.10.3. Einheitstest
- 3.10.4. Integrationstests
- 3.10.5. Validierungstest
- 3.10.6. Systemprüfung

Modul 4. Videospielkonsolen und -geräte

4.1. Geschichte der Videospielprogrammierung
- 4.1.1. Atari-Zeit (1977-1985)
- 4.1.2. NES und SNES Zeit (1985-1995)
- 4.1.3. PlayStation/PlayStation 2 Zeit (1995-2005)
- 4.1.4. Xbox 360, PS3 und Wii Zeit (2005-2013)
- 4.1.5. Xbox One, PS4 und Wii U-Switch Zeit (2013-heute)
- 4.1.6. Die Zukunft

4.2. Geschichte des Gameplays in Videospielen
- 4.2.1. Einführung
- 4.2.2. Sozialer Kontext
- 4.2.3. Strukturelles Diagramm
- 4.2.4. Zukunft

4.3. Anpassung an die moderne Zeit
- 4.3.1. Bewegungsbasierte Spiele
- 4.3.2. Virtuelle Realität
- 4.3.3. Erweiterte Reality
- 4.3.4. Gemischte Realität

4.4. Unity: *Scripting* I und Beispiele
- 4.4.1. Was ist ein *Script*?
- 4.4.2. Unser erstes *Script*
- 4.4.3. Hinzufügen eines *Scripts*
- 4.4.4. Öffnen eines *Scripts*
- 4.4.5. *MonoBehaviour*
- 4.4.6. *Debugging*

4.5. Unity: *Scripting* II und Beispiele
- 4.5.1. Tastatur- und Mauseingabe
- 4.5.2. *Raycast*
- 4.5.3. Instanziierung
- 4.5.4. Variablen
- 4.5.5. Öffentliche und serialisierte Variablen

4.6. Unity: *Scripting* III und Beispiele
- 4.6.1. Beschaffung von Komponenten
- 4.6.2. Komponenten modifizieren
- 4.6.3. Testen
- 4.6.4. Mehrere Objekte
- 4.6.5. *Colliders und Triggers*
- 4.6.6. Quaternionen

4.7. Peripheriegeräte
- 4.7.1. Entwicklung und Klassifizierung
- 4.7.2. Peripheriegeräte und Schnittstellen
- 4.7.3. Aktuelle Peripheriegeräte
- 4.7.4. Nahe Zukunft

4.8. Videospiele: Zukunftsperspektiven
- 4.8.1. Cloud-basiertes Spielen
- 4.8.2. Treiberlos
- 4.8.3. Immersive Realität
- 4.8.4. Andere Alternativen

4.9. Architektur
- 4.9.1. Besondere Anforderungen für Videospiele
- 4.9.2. Entwicklung der Architektur
- 4.9.3. Zeitgenössische Architektur
- 4.9.4. Unterschiede zwischen den Architekturen

4.10. Entwicklungskits und ihre Entwicklung
- 4.10.1. Einführung
- 4.10.2. Entwicklungskits der dritten Generation
- 4.10.3. Entwicklungskits der vierten Generation
- 4.10.4. Entwicklungskits der fünften Generation
- 4.10.5. Entwicklungskits der sechsten Generation

Modul 5. Softwareentwicklung

5.1. Einführung in die Softwareentwicklung und -modellierung
- 5.1.1. Die Natur der Software
- 5.1.2. Die Besonderheit von *Webapps*
- 5.1.3. Softwareentwicklung
- 5.1.4. Der Software-Prozess
- 5.1.5. Die Praxis der Softwareentwicklung
- 5.1.6. Software-Mythen
- 5.1.7. Wie beginnt alles?
- 5.1.8. Objektorientierte Konzepte
- 5.1.9. Einführung in UML

5.2. Der Software-Prozess
- 5.2.1. Ein allgemeines Prozessmodell
- 5.2.2. Vorgeschriebene Prozessmodelle
- 5.2.3. Spezialisierte Prozessmodelle
- 5.2.4. Der vereinheitlichte Prozess
- 5.2.5. Personal- und Teamprozessmodelle
- 5.2.6. Was ist Agilität?
- 5.2.7. Was ist ein agiler Prozess?
- 5.2.8. *Scrum*
- 5.2.9. Werkzeugkasten für agile Prozesse

5.3. Grundsätze für die Praxis der Softwareentwicklung
- 5.3.1. Leitprinzipien des Prozesses
- 5.3.2. Prinzipien als Leitfaden für die Praxis
- 5.3.3. Grundsätze der Kommunikation
- 5.3.4. Grundsätze der Planung
- 5.3.5. Grundsätze der Modellierung
- 5.3.6. Konstruktionsprinzipien
- 5.3.7. Grundsätze für die Einführung

5.4. Verständnis der Anforderungen
- 5.4.1. Anforderungsmanagement
- 5.4.2. Schaffung der Grundlagen
- 5.4.3. Bedarfsermittlung
- 5.4.4. Entwicklung von Anwendungsfällen
- 5.4.5. Ausarbeitung des Anforderungsmodells
- 5.4.6. Aushandeln von Anforderungen
- 5.4.7. Validierung der Anforderungen

5.5. Anforderungsmodellierung: Szenarien, Informations- und Analyseklassen
- 5.5.1. Analyse der Anforderungen
- 5.5.2. Szenario-basiertes Modell
- 5.5.3. UML-Modelle, die den Anwendungsfall liefern
- 5.5.4. Konzepte der Datenmodellierung
- 5.5.5. Klassen-basiertes Modell
- 5.5.6. Klassendiagramme

5.6. Anforderungsmodellierung: Fluss, Verhalten und Muster
- 5.6.1. Anforderungen die die Strategien gestalten
- 5.6.2. Flussorientierte Modellierung
- 5.6.3. Zustandsdiagramme
- 5.6.4. Erstellung eines Verhaltensmodells
- 5.6.5. Sequenzdiagramme
- 5.6.6. Kommunikationsdiagramme
- 5.6.7. Muster für die Modellierung von Anforderungen

5.7. Konzepte des Designs
- 5.7.1. Design im Kontext der Softwareentwicklung
- 5.7.2. Der Entwurfsprozess
- 5.7.3. Konzepte des Designs
- 5.7.4. Objektorientierte Konzepte des Designs
- 5.7.5. Das Designmodell

5.8. Design der Architektur
- 5.8.1. Software-Architektur
- 5.8.2. Architektonische Gattungen
- 5.8.3. Architektonische Stile
- 5.8.4. Architektonischer Design
- 5.8.5. Entwicklung von alternativen Designs für die Architektur
- 5.8.6. Abbildung der Architektur mit Hilfe von Datenflüssen

5.9. Design auf Komponentenebene und musterbasierter Entwurf
- 5.9.1. Was ist eine Komponente?
- 5.9.2. Klassenbasiertes Komponentendesign
- 5.9.3. Verwirklichung des Designs auf Komponentenebene
- 5.9.4. Design der traditionellen Komponenten
- 5.9.5. Komponentenbasierte Entwicklung
- 5.9.6. Entwurfsmuster
- 5.9.7. Musterbasiertea Softwaredesign
- 5.9.8. Architektonische Muster
- 5.9.9. Musterdesign auf Komponentenebene
- 5.9.10. Musterdesign für Benutzeroberflächen

5.10. Softwarequalität und Projektmanagement
- 5.10.1. Qualität
- 5.10.2. Softwarequalität
- 5.10.3. Das Dilemma der Softwarequalität
- 5.10.4. Erreichen von Softwarequalität
- 5.10.5. Qualitätssicherung der Software
- 5.10.6. Das administrative Spektrum
- 5.10.7. Personal
- 5.10.8. Das Produkt
- 5.10.9. Der Prozess
- 5.10.10. Das Projekt
- 5.10.11. Grundsätze und Praktiken

Modul 6. Videospiel-Engines

6.1. Videospiele und IKTs
- 6.1.1. Einführung
- 6.1.2. Gelegenheiten
- 6.1.3. Herausforderungen
- 6.1.4. Schlussfolgerungen

6.2. Geschichte der Spiel-Engines
- 6.2.1. Einführung
- 6.2.2. Atari-Ära
- 6.2.3. 1980er Ära
- 6.2.4. Erste Motoren. 90er Jahre Ära
- 6.2.5. Aktuelle Motoren

6.3. Videospiel-Engines
- 6.3.1. Typen von Motoren
- 6.3.2. Teile einer Videospiel-Engine
- 6.3.3. Aktuelle Motoren
- 6.3.4. Auswahl eines Motors für unser Projekt

6.4. *Motor Game Maker*
- 6.4.1. Einführung
- 6.4.2. Entwurf eines Szenarios
- 6.4.3. *Sprites* und Animationen
- 6.4.4. Kollisionen
- 6.4.5. *Scripting* in GML

6.5. Motor Unreal Engine 4: Einführung
- 6.5.1. Was ist die Unreal Engine 4? Was ist ihre Philosophie?
- 6.5.2. Materialien
- 6.5.3. UI
- 6.5.4. Animationen
- 6.5.5. Partikel System
- 6.5.6. Künstliche Intelligenz
- 6.5.7. FPS

6.6. Motor Unreal Engine 4: *Visual Scripting*
- 6.6.1. *Blueprint*-Philosophie und *Visual Scripting*
- 6.6.2. *Debugging*
- 6.6.3. Arten von Variablen
- 6.6.4. Grundlegende Flusskontrolle

6.7. Motor Unity 5
- 6.7.1. Programmieren in C# und Visual Studio
- 6.7.2. Erschaffen von *Prefabs*
- 6.7.3. Verwendung von *Gizmos* zur Steuerung von Videospielen
- 6.7.4. Adaptiver Motor: 2D und 3D

6.8. Godot-Motor
- 6.8.1. Godots Designphilosophie
- 6.8.2. Objektorientiertes Design und Komposition
- 6.8.3. All-in-one-Paket
- 6.8.4. Freie und von der Gemeinschaft betriebene Software

6.9. RPG Maker-Engine
- 6.9.1. RPG Maker Philosophie
- 6.9.2. Als Bezug nehmen
- 6.9.3. Ein Spiel mit Persönlichkeit schaffen
- 6.9.4. Erfolgreiche kommerzielle Spiele

6.10. Motor Source 2
- 6.10.1. Source 2 Philosophie
- 6.10.2. Source und Source 2: Entwicklung
- 6.10.3. Nutzung der Community: audiovisuelle Inhalte und Videospiele
- 6.10.4. Die Zukunft der Source 2 Engine
- 6.10.5. *Mods* und erfolgreiche Spiele

Modul 7. Intelligente Systeme

7.1. Agententheorie
- 7.1.1. Geschichte des Konzepts
- 7.1.2. Definition von Agent
- 7.1.3. Agenten in der künstlichen Intelligenz
- 7.1.4. Agenten in der Softwareentwicklung

7.2. Agent-Architekturen
- 7.2.1. Der Denkprozess eines Agenten
- 7.2.2. Reaktive Agenten
- 7.2.3. Deduktive Agenten
- 7.2.4. Hybride Agenten
- 7.2.5. Vergleich

7.3. Informationen und Wissen
- 7.3.1. Unterscheidung zwischen Daten, Informationen und Wissen
- 7.3.2. Bewertung der Datenqualität
- 7.3.3. Methoden der Datenerfassung
- 7.3.4. Methoden der Informationsbeschaffung
- 7.3.5. Methoden zum Wissenserwerb

7.4. Wissensrepräsentation
- 7.4.1. Die Bedeutung der Wissensrepräsentation
- 7.4.2. Definition der Wissensrepräsentation durch ihre Rollen
- 7.4.3. Merkmale einer Wissensrepräsentation

7.5. Ontologien
- 7.5.1. Einführung in Metadaten
- 7.5.2. Philosophisches Konzept der Ontologie
- 7.5.3. Computergestütztes Konzept der Ontologie
- 7.5.4. Bereichsontologien und Ontologien auf höherer Ebene
- 7.5.5. Wie erstellt man eine Ontologie?

7.6. Ontologiesprachen und Software für die Erstellung von Ontologien
- 7.6.1. RDF-Tripel, Turtle und N3
- 7.6.2. RDF-Schema
- 7.6.3. OWL
- 7.6.4. SPARQL
- 7.6.5. Einführung in die verschiedenen Tools für die Erstellung von Ontologien
- 7.6.6. Installation und Verwendung von Protégé

7.7. Das semantische Web
- 7.7.1. Der aktuelle Stand und die Zukunft des semantischen Webs
- 7.7.2. Anwendungen des semantischen Webs

7.8. Andere Modelle der Wissensdarstellung
- 7.8.1. Wortschatz
- 7.8.2. Globale Sicht
- 7.8.3. Taxonomie
- 7.8.4. Thesauri
- 7.8.5. Folksonomien
- 7.8.6. Vergleich
- 7.8.7. *Mind Map*

7.9. Bewertung und Integration von Wissensrepräsentationen
- 7.9.1. Logik nullter Ordnung
- 7.9.2. Logik erster Ordnung
- 7.9.3. Beschreibende Logik
- 7.9.4. Beziehung zwischen verschiedenen Arten von Logik
- 7.9.5. Prolog: Programmierung auf Basis der Logik erster Ordnung

7.10. Semantische *Reasoner*, wissensbasierte Systeme und Expertensysteme
- 7.10.1. Konzept des *Reasoners*
- 7.10.2. Anwendungen eines *Reasoners*
- 7.10.3. Wissensbasierte Systeme
- 7.10.4. MYCIN, Geschichte der Expertensysteme
- 7.10.5. Elemente und Architektur von Expertensystemen
- 7.10.6. Erstellung von Expertensystemen

Modul 8. Programmierung in Echtzeit

8.1. Grundlegende Konzepte der parallelen Programmierung
- 8.1.1. Grundlegende Konzepte
- 8.1.2. Parallelität
- 8.1.3. Vorteile der Parallelität
- 8.1.4. Parallelität und Hardware

8.2. Grundlegende Strukturen zur Unterstützung der Parallelität in Java
- 8.2.1. Parallelität in Java
- 8.2.2. *Threads* erstellen
- 8.2.3. Methoden
- 8.2.4. Synchronisierung

8.3. *Threads*, Lebenszyklus, Prioritäten, Unterbrechungen, Zustände, *Executors*
- 8.3.1. *Threads*
- 8.3.2. Lebenszyklus
- 8.3.3. Prioritäten
- 8.3.4. Unterbrechungen
- 8.3.5. Zustände
- 8.3.6. *Executors*

8.4. Gegenseitiger Ausschluss
- 8.4.1. Was bedeutet gegenseitiger Ausschluss?
- 8.4.2. Dekkers Algorithmus
- 8.4.3. Petersons Algorithmus
- 8.4.4. Gegenseitiger Ausschluss in Java

8.5. Abhängigkeiten vom Zustand
- 8.5.1. Injektion von Abhängigkeiten
- 8.5.2. Java-Implementierung des Musters
- 8.5.3. Wege zur Injektion von Abhängigkeiten
- 8.5.4. Beispiel

8.6. Entwurfsmuster
- 8.6.1. Einführung
- 8.6.2. Erzeugungsmuster
- 8.6.3. Struktur-Muster
- 8.6.4. Verhaltensmuster

8.7. Verwendung von Java-Bibliotheken
- 8.7.1. Was sind Bibliotheken in Java?
- 8.7.2. Mockito-all, mockito-core
- 8.7.3. Guava
- 8.7.4. Commons-io
- 8.7.5. Commons-lang, commons-lang3

8.8. *Shaders*-Programmierung
- 8.8.1. 3D-Pipeline und Raster
- 8.8.2. Vertex Shading
- 8.8.3. Pixel Shading: Beleuchtung I
- 8.8.4. Pixel Shading: Beleuchtung II
- 8.8.5. Post-Effekte

8.9. Programmierung in Echtzeit
- 8.9.1. Einführung
- 8.9.2. Verarbeitung von Unterbrechungen
- 8.9.3. Synchronisierung und Kommunikation zwischen Prozessen
- 8.9.4. Planungssysteme in Echtzeit

8.10. Planung in Echtzeit
- 8.10.1. Konzepte
- 8.10.2. Referenzmodell für Echtzeitsysteme
- 8.10.3. Planungspolitik
- 8.10.4. Zyklische Planer
- 8.10.5. Planer mit statischen Eigenschaften
- 8.10.6. Planer mit dynamischen Eigenschaften

Modul 9. Design und Entwicklung von Webspielen

9.1. Ursprünge und Standards des Webs
- 9.1.1. Die Ursprünge des Internets
- 9.1.2. Die Entstehung des *World Wide Web*
- 9.1.3. Aufkommen von Webstandards
- 9.1.4. Der Aufstieg der Webstandards

9.2. HTTP und Client-Server-Struktur
- 9.2.1. Client-Server-Rolle
- 9.2.2. Client-Server-Kommunikation
- 9.2.3. Jüngste Geschichte
- 9.2.4. Zentralisierte Datenverarbeitung

9.3. Webprogrammierung: Einführung
- 9.3.1. Grundlegende Konzepte
- 9.3.2. Einrichten eines Webservers
- 9.3.3. HTML5 Grundlagen
- 9.3.4. HTML-Formulare

9.4. Einführung in HTML und Beispiele
- 9.4.1. Geschichte von HTML5
- 9.4.2. Elemente von HTML5
- 9.4.3. APIS
- 9.4.4. CCS3

9.5. *Document Object Model*
- 9.5.1. Was ist das *Document Object Model*?
- 9.5.2. Verwendung von *DOCTYPE*
- 9.5.3. Die Bedeutung der Validierung von HTML
- 9.5.4. Zugriff auf Elemente
- 9.5.5. Elemente und Text erstellen
- 9.5.6. innerHTML verwenden
- 9.5.7. Ein Textelement oder einen Knoten löschen
- 9.5.8. Lesen und Schreiben der Attribute eines Elements
- 9.5.9. Manipulation von Elementstilen
- 9.5.10. Mehrere Dateien auf einmal anhängen

9.6. Einführung in CSS und Beispiele
- 9.6.1. CSS3-Syntax
- 9.6.2. Stil-Blätter
- 9.6.3. *Tags*
- 9.6.4. Selektoren
- 9.6.5. Webgestaltung mit CSS

9.7. Einführung in JavaScript und Beispiele
- 9.7.1. Was ist JavaScript?
- 9.7.2. Kurze Geschichte der Sprache
- 9.7.3. JavaScript-Versionen
- 9.7.4. Ein Dialogfeld anzeigen
- 9.7.5. JavaScript-Syntax
- 9.7.6. Skripte verstehen
- 9.7.7. Räume
- 9.7.8. Kommentare
- 9.7.9. Funktionen
- 9.7.10. Seiteninternes und externes JavaScript

9.8. Funktionen in JavaScript
- 9.8.1. Funktionsdeklarationen
- 9.8.2. Funktion Ausdrücke
- 9.8.3. Funktionen aufrufen
- 9.8.4. Rekursion
- 9.8.5. Verschachtelte Funktionen und Schließungen
- 9.8.6. Variable Konservierung
- 9.8.7. Mehrfach verschachtelte Funktionen
- 9.8.8. Namenskonflikte
- 9.8.9. Schließungen
- 9.8.10. Parameter einer Funktion

9.9. PlayCanvas für die Entwicklung von Webspielen
- 9.9.1. Was ist PlayCanvas?
- 9.9.2. Projekt-Konfiguration
- 9.9.3. Ein Objekt erstellen
- 9.9.4. Hinzufügen von Physik
- 9.9.5. Hinzufügen eines Modells
- 9.9.6. Ändern der Schwerkraft- und Szeneneinstellungen
- 9.9.7. *Scripts* ausführen
- 9.9.8. Kamera-Steuerungen

9.10. Phaser für die Entwicklung von Webspielen
- 9.10.1. Was ist Phaser?
- 9.10.2. Ressourcen laden
- 9.10.3. Die Welt bauen
- 9.10.4. Plattformen
- 9.10.5. Der Spieler
- 9.10.6. Hinzufügen von Physik
- 9.10.7. Verwendung der Tastatur
- 9.10.8. Aufnehmen von *Pickups*
- 9.10.9. Punkte und Wertung
- 9.10.10. Springende Bomben

Modul 10. Multiplayer-Netzwerke und -Systeme

10.1. Geschichte und Entwicklung von Multiplayer-Spielen
- 10.1.1. 1970er Jahre: erste Multiplayer-Spiele
- 10.1.2. 1990er Jahre: Duke Nukem, Doom, Quake
- 10.1.3. Der Aufstieg der Multiplayer-Videospiele
- 10.1.4. Lokaler und Online-Multiplayer
- 10.1.5. Partyspiele

10.2. Multiplayer-Geschäftsmodelle
- 10.2.1. Entstehung und Funktionsweise von neuen Geschäftsmodellen
- 10.2.2. Online-Verkaufsdienstleistungen
- 10.2.3. Frei zum Spielen
- 10.2.4. Micropayments
- 10.2.5. Werbung
- 10.2.6. Abonnement mit monatlichen Zahlungen
- 10.2.7. Pay-per-play
- 10.2.8. Testen vor dem Kauf

10.3. Lokale Spiele und vernetzte Spiele
- 10.3.1. Lokale Spiele: Erste Schritte
- 10.3.2. Partyspiele: Nintendo und Familienzusammengehörigkeit
- 10.3.3. Netzwerkspiele: Anfänge
- 10.3.4. Entwicklung von Netzwerkspielen

10.4. OSI-Modell: Schichten I
- 10.4.1. OSI-Modell: Einleitung
- 10.4.2. Physikalische Schicht
- 10.4.3. Datenübertragungsschicht
- 10.4.4. Netzwerkschicht

10.5. OSI-Modell: Schichten II
- 10.5.1. Transportschicht
- 10.5.2. Sitzungsschicht
- 10.5.3. Präsentationsschicht
- 10.5.4. Anwendungsschicht

10.6. Computernetzwerke und das Internet
- 10.6.1. Was ist ein Computernetzwerk?
- 10.6.2. Software
- 10.6.3. Hardware
- 10.6.4. Server
- 10.6.5. Netzwerkspeicher
- 10.6.6. Netzwerk-Protokolle

10.7. Mobile und drahtlose Netzwerke
- 10.7.1. Mobiles Netzwerk
- 10.7.2. Drahtloses Netzwerk
- 10.7.3. Betrieb von mobilen Netzwerken
- 10.7.4. Digitale Technologie

10.8. Sicherheit
- 10.8.1. Persönliche Sicherheit
- 10.8.2. *Hacks* und *Cheats* in Videospielen
- 10.8.3. Anti-Fallen-Sicherheit
- 10.8.4. Analyse von Sicherheitssystemen gegen Betrug

10.9. Mehrspielersysteme: Server
- 10.9.1. *Server Hosting*
- 10.9.2. MMO-Videospiele
- 10.9.3. Dedizierte Videospiel-Server
- 10.9.4. *LAN Parties*

10.10. Design und Programmierung von Multiplayer-Videospielen
- 10.10.1. Grundlagen der Entwicklung von Multiplayer-Spielen in Unreal
- 10.10.2. Grundlagen der Entwicklung von Multiplayer-Spielen in Unity
- 10.10.3. Wie gestaltet man ein Multiplayer-Spiel unterhaltsam?
- 10.10.4. Jenseits eines Controllers: Innovation in der Multiplayer-Steuerung

Dank der Relearning-Methode, die auf der Wiederholung der wichtigsten Konzepte basiert, können Sie die langen Lernzeiten reduzieren"

06

Praktikum

Nachdem der Videospielprofi die theoretische Online-Fortbildung abgeschlossen hat, beinhaltet dieser Abschluss ein Praktikum in einem Kreativstudio und bei einem Videospielentwickler. Auf diese Weise kommen die Studenten in direkten Kontakt mit Entwicklerteams in der Branche und können alles, was sie gelernt haben, unter Beweis stellen. Während dieser Lernphase werden die Studenten von einem Tutor betreut, der sie bis zum Ende des Kurses begleitet.

Arbeiten Sie während der praktischen Phase dieses Kurses neben großen Entwicklern und in einem Team“

Das Praktikum im Rahmen dieses Programms in Programmierung von Videospielen dauert 3 Wochen. Die Studenten gehen von Montag bis Freitag in das Kreativstudio und zum Videospielentwickler, wo sie an jedem Tag 8 aufeinanderfolgende Stunden praktischen Unterricht von einem Spezialisten erhalten.

In dieser Praxisphase können die Studenten ihr gesamtes in diesem Blended-Learning-Masterstudiengang erworbenes Wissen über Programmierung anwenden. Gemeinsam mit anderen Fachleuten aus diesem Unternehmen können sie sich mit dem Rest des Teams abstimmen und die wichtigsten Programmierwerkzeuge und -software erlernen. Auf diese Weise erhalten sie eine Lernerfahrung, die der Realität der Videospielindustrie sehr nahe kommt.

Während des Aufenthalts in diesem Studio wird die Fachkraft nicht nur die übliche Programmiersprache verwenden, sondern auch in der Lage sein, Multiplayer-Videospiele zu entwickeln. Eine Kategorie von Videospielen, die auf dem Vormarsch ist und die *Gamer* fasziniert.

Der praktische Unterricht wird unter aktiver Beteiligung des Studenten durchgeführt, der die Aktivitäten und Verfahren der einzelnen Kompetenzbereiche ausführt (lernen, zu lernen und zu tun), mit der Begleitung und Anleitung von Dozenten und anderen Ausbildungskollegen, die die Teamarbeit und die multidisziplinäre

> *Bilden Sie sich in einem Studio aus, das Ihnen die Möglichkeit gibt, Ihr Potenzial als Videospielprogrammierer unter Beweis zu stellen“*

Integration als transversale Kompetenzen für die Programmierpraxis erleichtern (lernen, zu sein und lernen, mit anderen in Beziehung zu treten).

Die im Folgenden beschriebenen Verfahren bilden die Grundlage für den praktischen Teil der Ausbildung. Ihre Durchführung hängt von der Verfügbarkeit und Arbeitsbelastung des Zentrums ab:

Modul	Praktische Tätigkeit	Anzahl
Erstellung von Datenstrukturen und Algorithmen	Durchführen der *Backtracking*-Technik für die Erstellung von Daten und Algorithmen	1
	Mitwirken an der Analyse von Algorithmen zur Effizienzsteigerung	
	Durchführen von Aufgaben zur Messung der Eingabegröße und der Laufzeit	
	Erstellen von Sortieralgorithmen mit Bäumen, mit *Heaps*, mit Graphen und mit *Greedy*	
Objektorientierte Programmierung	Verwenden des *Factory Pattern, Singleton Pattern, Observer Pattern* und *Composite Pattern* bei der Erstellung von Objekten	1
	Erstellen, Erfassen und Verwalten von Ausnahmen bei der Objekterstellung	
	Durchführen nebenläufiger Programmierung	
	Verwenden von Sperr- und Kommunikationsmechanismen	
	Erstellen von Softwaredokumentation und -tests	
Programmierung in Echtzeit	Erstellen und Synchronisieren von *Threads*	1
	Programmieren von *Shadern*	
	Implementieren des Musters in Java und Verwenden von Java-Bibliotheken	
	Erstellen von Post-Effekten	
	Verarbeiten von Unterbrechungen, Synchronisierung und Kommunikation zwischen Prozessen	
Design und Entwicklung von Webspielen	Programmieren des Webs mit HTML-Formularen	1
	Verwenden von DOCTYPE und innerHTML zur Entwicklung von Webspielen	
	Verwenden von PlayCanvas zur Entwicklung von Webspielen	
	Einrichten eines Projekts zur Gestaltung und Entwicklung von Webspielen	

Zivile Haftpflichtversicherung

Das Hauptanliegen dieser Einrichtung ist es, die Sicherheit sowohl der Fachkräfte im Praktikum als auch der anderen am Praktikum beteiligten Personen im Unternehmen zu gewährleisten. Zu den Maßnahmen, mit denen dies erreicht werden soll, gehört auch die Reaktion auf Zwischenfälle, die während des gesamten Lehr- und Lernprozesses auftreten können.

Zu diesem Zweck verpflichtet sich diese Bildungseinrichtung, eine Haftpflichtversicherung abzuschließen, die alle Eventualitäten abdeckt, die während des Aufenthalts im Praktikumszentrum auftreten können.

Diese Haftpflichtversicherung für die Fachkräfte im Praktikum hat eine umfassende Deckung und wird vor Beginn der Praktischen Ausbildung abgeschlossen. Auf diese Weise muss sich der Berufstätige keine Sorgen machen, wenn er mit einer unerwarteten Situation konfrontiert wird, und ist bis zum Ende des praktischen Programms in der Einrichtung abgesichert

Allgemeine Bedingungen der Praktischen Ausbildung

Die allgemeinen Bedingungen des Praktikumsvertrags für das Programm lauten wie folgt:

1. BETREUUNG: Während der Praktischen Ausbildung werden dem Studenten zwei Tutoren zugeteilt, die ihn während des gesamten Prozesses begleiten und alle Zweifel und Fragen klären, die auftauchen können. Einerseits gibt es einen professionellen Tutor des Praktikumszentrums, der die Aufgabe hat, den Studenten zu jeder Zeit zu begleiten und zu unterstützen. Andererseits wird dem Studenten auch ein akademischer Tutor zugewiesen, dessen Aufgabe es ist, den Studenten während des gesamten Prozesses zu koordinieren und zu unterstützen, Zweifel zu beseitigen und ihm alles zu erleichtern, was er braucht. Auf diese Weise wird die Fachkraft begleitet und kann alle Fragen stellen, die sie hat, sowohl praktischer als auch akademischer Natur.

2. DAUER: Das Praktikumsprogramm umfasst drei zusammenhängende Wochen praktischer Ausbildung in 8-Stunden-Tagen an fünf Tagen pro Woche. Die Anwesenheitstage und der Stundenplan liegen in der Verantwortung des Zentrums und die Fachkraft wird rechtzeitig darüber informiert, damit sie sich organisieren kann.

3. NICHTERSCHEINEN: Bei Nichterscheinen am Tag des Beginns der Praktischen Ausbildung verliert der Student den Anspruch auf diese ohne die Möglichkeit einer Rückerstattung oder der Änderung der Daten. Eine Abwesenheit von mehr als zwei Tagen vom Praktikum ohne gerechtfertigten/medizinischen Grund führt zum Rücktritt vom Praktikum und damit zu seiner automatischen Beendigung. Jedes Problem, das im Laufe des Praktikums auftritt, muss dem akademischen Tutor ordnungsgemäß und dringend mitgeteilt werden.

4. ZERTIFIZIERUNG: Der Student, der die Praktische Ausbildung bestanden hat, erhält ein Zertifikat, das den Aufenthalt in dem betreffenden Zentrum bestätigt.

5. ARBEITSVERHÄLTNIS: Die Praktische Ausbildung begründet kein Arbeitsverhältnis irgendeiner Art.

6. VORBILDUNG: Einige Zentren können für die Teilnahme an der Praktischen Ausbildung eine Bescheinigung über ein vorheriges Studium verlangen. In diesen Fällen muss sie der TECH-Praktikumsabteilung vorgelegt werden, damit die Zuweisung des gewählten Zentrums bestätigt werden kann.

7. NICHT INBEGRIFFEN: Die Praktische Ausbildung beinhaltet keine Elemente, die nicht in diesen Bedingungen beschrieben sind. Daher sind Unterkunft, Transport in die Stadt, in der das Praktikum stattfindet, Visa oder andere nicht beschriebene Leistungen nicht inbegriffen.

Der Student kann sich jedoch an seinen akademischen Tutor wenden, wenn er Fragen hat oder Empfehlungen in dieser Hinsicht erhalten möchte. Dieser wird ihm alle notwendigen Informationen geben, um die Verfahren zu erleichtern.

07

Studienmethodik

TECH ist die erste Universität der Welt, die die Methodik der ***case studies*** mit ***Relearning*** kombiniert, einem 100%igen Online-Lernsystem, das auf geführten Wiederholungen basiert.

Diese disruptive pädagogische Strategie wurde entwickelt, um Fachleuten die Möglichkeit zu bieten, ihr Wissen zu aktualisieren und ihre Fähigkeiten auf intensive und gründliche Weise zu entwickeln. Ein Lernmodell, das den Studenten in den Mittelpunkt des akademischen Prozesses stellt und ihm die Hauptrolle zuweist, indem es sich an seine Bedürfnisse anpasst und die herkömmlichen Methoden beiseite lässt.

TECH bereitet Sie darauf vor, sich neuen Herausforderungen in einem unsicheren Umfeld zu stellen und in Ihrer Karriere erfolgreich zu sein“

Der Student: die Priorität aller Programme von TECH

Bei der Studienmethodik von TECH steht der Student im Mittelpunkt.
Die pädagogischen Instrumente jedes Programms wurden unter Berücksichtigung der Anforderungen an Zeit, Verfügbarkeit und akademische Genauigkeit ausgewählt, die heutzutage nicht nur von den Studenten, sondern auch von den am stärksten umkämpften Stellen auf dem Markt verlangt werden.

Beim asynchronen Bildungsmodell von TECH entscheidet der Student selbst, wie viel Zeit er mit dem Lernen verbringt und wie er seinen Tagesablauf gestaltet, und das alles bequem von einem elektronischen Gerät seiner Wahl aus. Der Student muss nicht an Präsenzveranstaltungen teilnehmen, die er oft nicht wahrnehmen kann.
Die Lernaktivitäten werden nach eigenem Ermessen durchgeführt. Er kann jederzeit entscheiden, wann und von wo aus er lernen möchte.

Bei TECH gibt es KEINE Präsenzveranstaltungen (an denen man nie teilnehmen kann)“

Die international umfassendsten Lehrpläne

TECH zeichnet sich dadurch aus, dass sie die umfassendsten Studiengänge im universitären Umfeld anbietet. Dieser Umfang wird durch die Erstellung von Lehrplänen erreicht, die nicht nur die wesentlichen Kenntnisse, sondern auch die neuesten Innovationen in jedem Bereich abdecken.

Durch ihre ständige Aktualisierung ermöglichen diese Programme den Studenten, mit den Veränderungen des Marktes Schritt zu halten und die von den Arbeitgebern am meisten geschätzten Fähigkeiten zu erwerben. Auf diese Weise erhalten die Studenten, die ihr Studium bei TECH absolvieren, eine umfassende Vorbereitung, die ihnen einen bedeutenden Wettbewerbsvorteil verschafft, um in ihrer beruflichen Laufbahn voranzukommen.

Und das von jedem Gerät aus, ob PC, Tablet oder Smartphone.

> *Das Modell der TECH ist asynchron, d. h. Sie können an Ihrem PC, Tablet oder Smartphone studieren, wo immer Sie wollen, wann immer Sie wollen und so lange Sie wollen“*

Case studies oder Fallmethode

Die Fallmethode ist das am weitesten verbreitete Lernsystem an den besten Wirtschaftshochschulen der Welt. Sie wurde 1912 entwickelt, damit Studenten der Rechtswissenschaften das Recht nicht nur auf der Grundlage theoretischer Inhalte erlernten, sondern auch mit realen komplexen Situationen konfrontiert wurden. Auf diese Weise konnten sie fundierte Entscheidungen treffen und Werturteile darüber fällen, wie diese zu lösen sind. Sie wurde 1924 als Standardlehrmethode in Harvard etabliert.

Bei diesem Lehrmodell ist es der Student selbst, der durch Strategien wie *Learning by doing* oder *Design Thinking*, die von anderen renommierten Einrichtungen wie Yale oder Stanford angewandt werden, seine berufliche Kompetenz aufbaut.

Diese handlungsorientierte Methode wird während des gesamten Studiengangs angewandt, den der Student bei TECH absolviert. Auf diese Weise wird er mit zahlreichen realen Situationen konfrontiert und muss Wissen integrieren, recherchieren, argumentieren und seine Ideen und Entscheidungen verteidigen. All dies unter der Prämisse, eine Antwort auf die Frage zu finden, wie er sich verhalten würde, wenn er in seiner täglichen Arbeit mit spezifischen, komplexen Ereignissen konfrontiert würde.

Relearning-Methode

Bei TECH werden die *case studies* mit der besten 100%igen Online-Lernmethode ergänzt: *Relearning*.

Diese Methode bricht mit traditionellen Lehrmethoden, um den Studenten in den Mittelpunkt zu stellen und ihm die besten Inhalte in verschiedenen Formaten zu vermitteln. Auf diese Weise kann er die wichtigsten Konzepte der einzelnen Fächer wiederholen und lernen, sie in einem realen Umfeld anzuwenden.

In diesem Sinne und gemäß zahlreicher wissenschaftlicher Untersuchungen ist die Wiederholung der beste Weg, um zu lernen. Aus diesem Grund bietet TECH zwischen 8 und 16 Wiederholungen jedes zentralen Konzepts innerhalb ein und derselben Lektion, die auf unterschiedliche Weise präsentiert werden, um sicherzustellen, dass das Wissen während des Lernprozesses vollständig gefestigt wird.

Das Relearning ermöglicht es Ihnen, mit weniger Aufwand und mehr Leistung zu lernen, sich mehr auf Ihre Spezialisierung einzulassen, einen kritischen Geist zu entwickeln, Argumente zu verteidigen und Meinungen zu kontrastieren: eine direkte Gleichung zum Erfolg.

Ein 100%iger virtueller Online-Campus mit den besten didaktischen Ressourcen

Um seine Methodik wirksam anzuwenden, konzentriert sich TECH darauf, den Studenten Lehrmaterial in verschiedenen Formaten zur Verfügung zu stellen: Texte, interaktive Videos, Illustrationen und Wissenskarten, um nur einige zu nennen. Sie alle werden von qualifizierten Lehrkräften entwickelt, die ihre Arbeit darauf ausrichten, reale Fälle mit der Lösung komplexer Situationen durch Simulationen, dem Studium von Zusammenhängen, die für jede berufliche Laufbahn gelten, und dem Lernen durch Wiederholung mittels Audios, Präsentationen, Animationen, Bildern usw. zu verbinden.

Die neuesten wissenschaftlichen Erkenntnisse auf dem Gebiet der Neurowissenschaften weisen darauf hin, dass es wichtig ist, den Ort und den Kontext, in dem der Inhalt abgerufen wird, zu berücksichtigen, bevor ein neuer Lernprozess beginnt. Die Möglichkeit, diese Variablen individuell anzupassen, hilft den Menschen, sich zu erinnern und Wissen im Hippocampus zu speichern, um es langfristig zu behalten. Dies ist ein Modell, das als *Neurocognitive context-dependent e-learning* bezeichnet wird und in diesem Hochschulstudium bewusst angewendet wird.

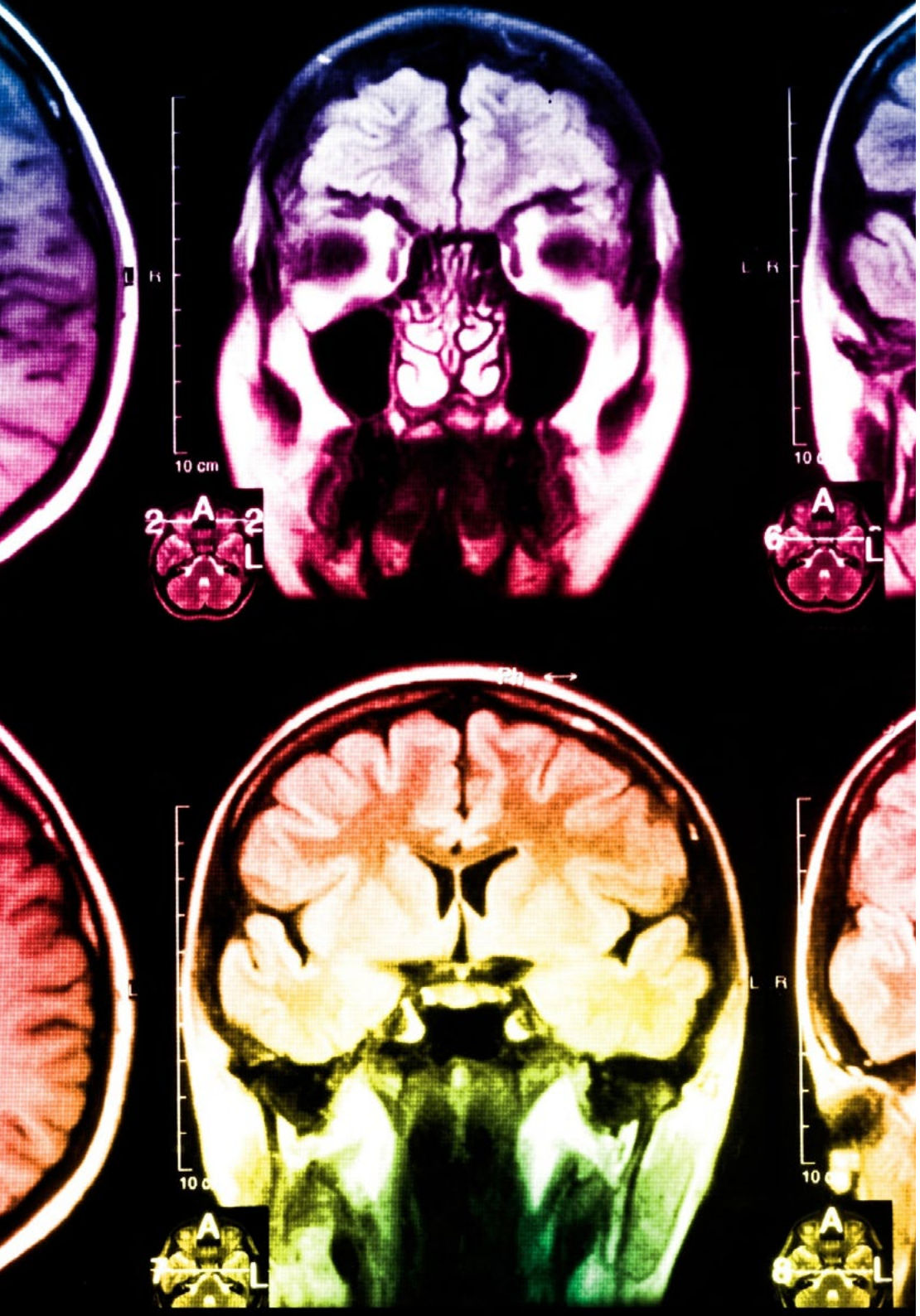
Zum anderen, auch um den Kontakt zwischen Mentor und Student so weit wie möglich zu begünstigen, wird eine breite Palette von Kommunikationsmöglichkeiten angeboten, sowohl in Echtzeit als auch zeitversetzt (internes Messaging, Diskussionsforen, Telefondienst, E-Mail-Kontakt mit dem technischen Sekretariat, Chat und Videokonferenzen).

Darüber hinaus wird dieser sehr vollständige virtuelle Campus den Studenten der TECH die Möglichkeit geben, ihre Studienzeiten entsprechend ihrer persönlichen Verfügbarkeit oder ihren beruflichen Verpflichtungen zu organisieren. Auf diese Weise haben sie eine globale Kontrolle über die akademischen Inhalte und ihre didaktischen Hilfsmittel, in Übereinstimmung mit ihrer beschleunigten beruflichen Weiterbildung.

> *Der Online-Studienmodus dieses Programms wird es Ihnen ermöglichen, Ihre Zeit und Ihr Lerntempo zu organisieren und an Ihren Zeitplan anzupassen“*

Die Wirksamkeit der Methode wird durch vier Schlüsselergebnisse belegt:

1. Studenten, die diese Methode anwenden, nehmen nicht nur Konzepte auf, sondern entwickeln auch ihre geistigen Fähigkeiten durch Übungen zur Bewertung realer Situationen und zur Anwendung ihres Wissens.
2. Das Lernen basiert auf praktischen Fähigkeiten, die es den Studenten ermöglichen, sich besser in die reale Welt zu integrieren.
3. Eine einfachere und effizientere Aufnahme von Ideen und Konzepten wird durch die Verwendung von Situationen erreicht, die aus der Realität entstanden sind.
4. Das Gefühl der Effizienz der investierten Anstrengung wird zu einem sehr wichtigen Anreiz für die Studenten, was sich in einem größeren Interesse am Lernen und einer Steigerung der Zeit, die für die Arbeit am Kurs aufgewendet wird, niederschlägt.



Die von ihren Studenten am besten bewertete Hochschulmethodik

Die Ergebnisse dieses innovativen akademischen Modells lassen sich an der Gesamtzufriedenheit der Absolventen der TECH ablesen.

Die Studenten bewerten die Qualität der Lehre, die Qualität der Materialien, die Kursstruktur und die Ziele als hervorragend. So überrascht es nicht, dass die Einrichtung von ihren Studenten auf der Bewertungsplattform Trustpilot mit 4,9 von 5 Punkten am besten bewertet wurde.

Sie können von jedem Gerät mit Internetanschluss (Computer, Tablet, Smartphone) auf die Studieninhalte zugreifen, da TECH in Sachen Technologie und Pädagogik führend ist.

Sie werden die Vorteile des Zugangs zu simulierten Lernumgebungen und des Lernens durch Beobachtung, d. h. Learning from an expert, nutzen können.

In diesem Programm stehen Ihnen die besten Lehrmaterialien zur Verfügung, die sorgfältig vorbereitet wurden:

Studienmaterial

Alle didaktischen Inhalte werden von den Fachkräfte, die den Kurs unterrichten werden, speziell für den Kurs erstellt, so dass die didaktische Entwicklung wirklich spezifisch und konkret ist.

Diese Inhalte werden dann auf ein audiovisuelles Format übertragen, das unsere Online-Arbeitsweise mit den neuesten Techniken ermöglicht, die es uns erlauben, Ihnen eine hohe Qualität in jedem der Stücke zu bieten, die wir Ihnen zur Verfügung stellen werden.

Übungen für Fertigkeiten und Kompetenzen

Sie werden Aktivitäten durchführen, um spezifische Kompetenzen und Fertigkeiten in jedem Fachbereich zu entwickeln. Übungen und Aktivitäten zum Erwerb und zur Entwicklung der Fähigkeiten und Fertigkeiten, die ein Spezialist im Rahmen der Globalisierung, in der wir leben, entwickeln muss.

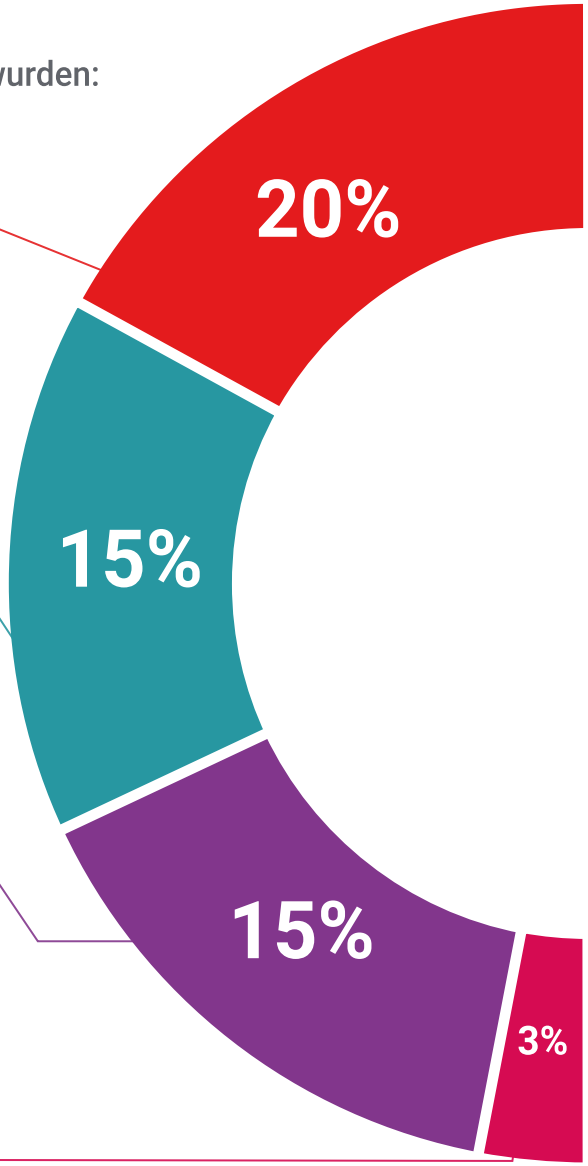
Interaktive Zusammenfassungen

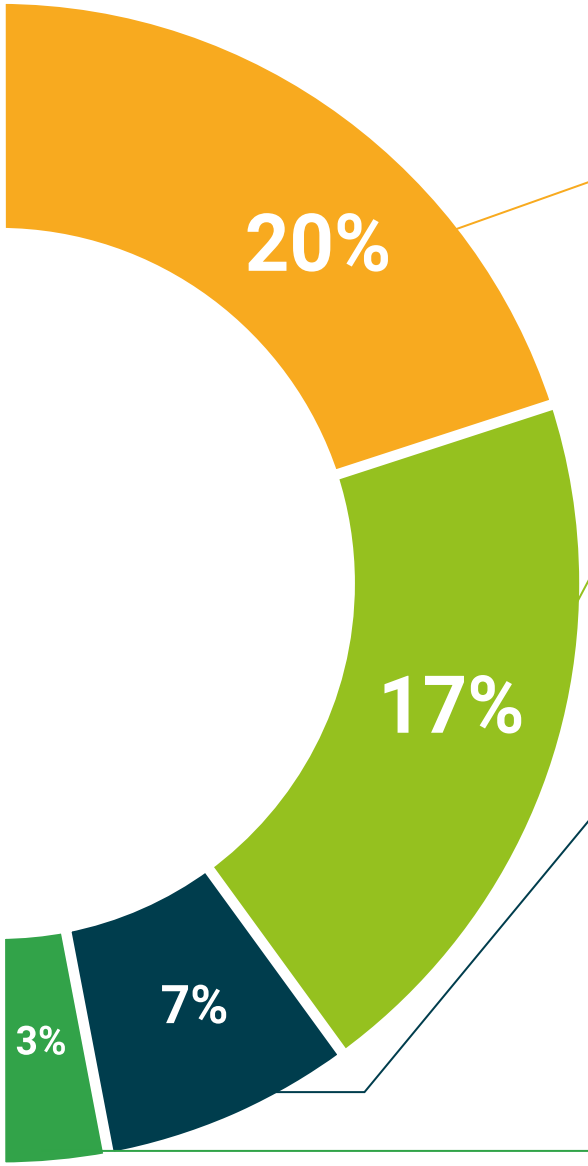
Wir präsentieren die Inhalte auf attraktive und dynamische Weise in multimedialen Pillen, Audios, Videos, Bilder, Diagramme und konzeptionelle Karten enthalten, um das Wissen zu festigen.

Dieses einzigartige System für die Präsentation multimedialer Inhalte wurde von Microsoft als „Europäische Erfolgsgeschichte" ausgezeichnet.

Weitere Lektüren

Aktuelle Artikel, Konsensdokumente, internationale Leitfäden... In unserer virtuellen Bibliothek haben Sie Zugang zu allem, was Sie für Ihre Ausbildung benötigen.





Case Studies

Sie werden eine Auswahl der besten *case studies* zu diesem Thema bearbeiten. Die Fälle werden von den besten Spezialisten der internationalen Szene präsentiert, analysiert und betreut.

Testing & Retesting

Während des gesamten Programms werden Ihre Kenntnisse in regelmäßigen Abständen getestet und wiederholt. Wir tun dies auf 3 der 4 Ebenen der Millerschen Pyramide.

Meisterklassen

Die Nützlichkeit der Expertenbeobachtung ist wissenschaftlich belegt.

Das sogenannte *Learning from an Expert* stärkt das Wissen und das Gedächtnis und schafft Vertrauen in unsere zukünftigen schwierigen Entscheidungen.

Kurzanleitungen zum Vorgehen

TECH bietet die wichtigsten Inhalte des Kurses in Form von Arbeitsblättern oder Kurzanleitungen an. Ein synthetischer, praktischer und effektiver Weg, um dem Studenten zu helfen, in seinem Lernen voranzukommen.

08

Qualifizierung

Dieser Blended-Learning-Masterstudiengang in Programmierung von Videospielen garantiert neben der präzisesten und aktuellsten Fortbildung auch den Zugang zu einem von der TECH Technologische Universität ausgestellten Diplom.

Schließen Sie dieses Programm erfolgreich ab und erhalten Sie Ihren Universitätsabschluss ohne lästige Reisen oder Formalitäten"

Dieser **Blended-Learning-Masterstudiengang in Programmierung von Videospielen** enthält das vollständigste und aktuellste Programm auf dem Markt.

Sobald der Student die Prüfungen bestanden hat, erhält er/sie per Post* mit Empfangsbestätigung das entsprechende Diplom, ausgestellt von der **TECH Technologischen Universität.**

Das von **TECH Technologische Universität** ausgestellte Diplom drückt die erworbene Qualifikation aus und entspricht den Anforderungen, die in der Regel von Stellenbörsen, Auswahlprüfungen und Berufsbildungsausschüssen verlangt werden.

Titel: **Blended-Learning-Masterstudiengang in Programmierung von Videospielen**

Modalität: **Blended Learning (Online + Praktikum)**

Dauer: **12 Monate**


tech technologische universität

Verleiht dieses

DIPLOM

an

Herr/Frau _____, mit Ausweis-Nr. _____

Für den erfolgreichen Abschluss und die Akkreditierung des Programms

SEMIPRÄSENTIELLER MASTERSTUDIENGANG

in

Programmierung von Videospielen

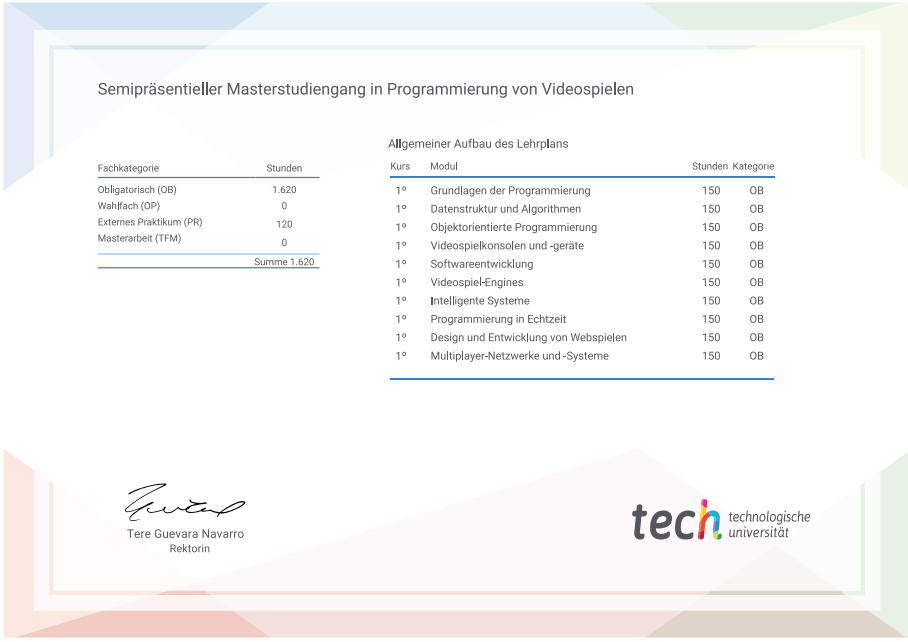
Es handelt sich um einen von dieser Universität verliehenen Abschluss, mit einer Dauer von 1.620 Stunden, mit Anfangsdatum tt/mm/jjjj und Enddatum tt/mm/jjjj.

TECH ist eine private Hochschuleinrichtung, die seit dem 28. Juni 2018 vom Ministerium für öffentliche Bildung anerkannt ist.

Zum 17. Juni 2020

Tere Guevara Navarro
Rektorin

einzigartiger Code TECH: AFWOR235 techtitute.com/titel


Semipräsentieller Masterstudiengang in Programmierung von Videospielen

Fachkategorie	Stunden
Obligatorisch (OB)	1.620
Wahlfach (OP)	0
Externes Praktikum (PR)	120
Masterarbeit (TFM)	0
	Summe 1.620

Allgemeiner Aufbau des Lehrplans

Kurs	Modul	Stunden	Kategorie
1º	Grundlagen der Programmierung	150	OB
1º	Datenstruktur und Algorithmen	150	OB
1º	Objektorientierte Programmierung	150	OB
1º	Videospielkonsolen und -geräte	150	OB
1º	Softwareentwicklung	150	OB
1º	Videospiel-Engines	150	OB
1º	Intelligente Systeme	150	OB
1º	Programmierung in Echtzeit	150	OB
1º	Design und Entwicklung von Webspielen	150	OB
1º	Multiplayer-Netzwerke und -Systeme	150	OB

Tere Guevara Navarro
Rektorin

tech technologische universität

*Haager Apostille. Für den Fall, dass der Student die Haager Apostille für sein Papierdiplom beantragt, wird TECH EDUCATION die notwendigen Vorkehrungen treffen, um diese gegen eine zusätzliche Gebühr zu beschaffen.

tech technologische universität

Blended-Learning-Masterstudiengang

Programmierung von Videospielen

Modalität: **Blended Learning (Online + Praktikum)**
Dauer: **12 Monate**
Qualifizierung: **TECH Technologische Universität**

Blended-Learning-Masterstudiengang
Programmierung von Videospielen

tech technologische universität