Asynchronous Module Definition

Mit der JavaScript-Programmschnittstelle AMD können Module, also Bausteine eines Softwaresystems, sowie deren Abhängigkeiten asynchron geladen werden.
Damit werden die Konzepte der Modularisierung und der Wiederverwertung, die sonst aus anderen Programmiersprachen wie beispielsweise JAVA bekannt sind, in JavaScript aufgegriffen und nutzbar gemacht. Die JavaScript Anwendung wird also in mehrere kleineren Module aufgeteilt, was diverse Vorteile bietet.

Das besondere an der AMD Vorgehensweise ist, dass Module gleich mit ihren Abhängigkeiten zusammengeladen werden, so wird verhindert, dass Module auf Code zugreifen, der noch gar nicht geladen wurde. Module haben eine eindeutige String-ID, über die sie definiert und geladen werden. Im Vergleich zu einigen anderen Ansätzen ist AMD leichter direkt im Browser zu debuggen, da es gängige Probleme die etwa bei der Nutzung von cdns oder cross-domain auftauchen, ausschließt.

AMD wird in verschiedenen Frameworks eingesetzt, unter anderem in RequireJS, dem Dojo-Toolkit und curl.js. Für ausführliche Informationen zur Implementation empfiehlt sich also ein Blick in die Dokumentation von RequireJS oder einem anderen der oben genannten Frameworks. Daher im Folgenden nur eine minimale Übersicht über die Schritte zur Implementation mit RequireJS:

Um AMD zu verwenden muss in der TS Config bei compilerOptions das module auf „amd“ gestellt werden:

Damit man AMD bzw. RequireJS auch im Browser verwenden kann, muss man es in der html-Datei einbinden. Mit RequireJS funktioniert das jedoch ein wenig anders als mit den klassischen script Tags:

```
<script data-main="scripts/main.js" src="scripts/require.js"></script>
```

RequireJS lädt allen Code aus einem bestimmten Verzeichnis, in diesem Fall aus dem scripts-Ordner, daher müssen sich auch alle Module in diesem Ordner befinden. Außerdem müssen alle Module in einer Datei geladen werden, da man auf diese Methode nur eine .js-Datei in die html-Datei einbinden kann. Der data-main Tag gibt also das Script an, in dem alle Module geladen werden, wohingegen der src Tag den Pfad zur benötigten require.js Datei angibt.

Ein Modul mit einer Abhängigkeit wird dann im Quellcode wie folgt definiert:

```
1. define('testmodul', ['dependency1'], function (dependency1) {
2. return function () {};
3. });
```

Das Modul testmodul wird mit der Abhängigkeit dependency1 definiert und eine Funktion angegeben.