Umsetzung prüfen RevisionID in IOHDACodecDevice ändern (PropertyInjector.kext)

Beitrag von „Brumbaer“ vom 17. Dezember 2018, 04:36

Wie gesagt eine Klasse kann alles mögliche sein z.B. ein Gruppenname, aber die Klasse die einen Service realisiert ist eine Klasse im Sinner der Object Orientierten Programmierung.

Ich habe über Stunden hinweg ohne auf Programmierkenntnisse zurückzugreifen versucht zu beschreiben was eine Klasse ist und ich bin gescheitert.

Letztendlich ist eine Klasse ein Programmstück, das Instanzen von sich erzeugen kann und definiert was die Instanzen können (Interface) und wie sie es tun (Implementation).

Da Klassen Programmstücke sind, werden sie von einem Programmierer erstellt, mit dem Ziel eine bestimmte Aufgabe zu erfüllen.

Andere Klassen oder Programmstücke können auf Instanzen einer Klasse zugreifen und mit ihnen arbeiten, denn wie das geht sagt ihnen das Klassen Interface.

Weiss man welcher Klasse eine Instanz angehört, weiss man wie man mit ihr arbeiten kann ohne zu wissen zu müssen wie sie es genau tut.

Es gibt Mechanismen in macos, die Treiber laden. Die Treiber sind Instanzen von Treiberklassen.

Die Instanzen werden erzeugt und melden dann dem Betriebssystem, dass sie da sind.

Hat man nun ein Gerät, dass eine Instanz einer Klasse lädt, die macos nicht kannte als es erstellt wurde, sieht man alt aus, denn macos kennt ja das Klasseninterface nicht, da es die Klasse noch nicht gab als es erstellt wurde.

Da kommt die Vererbung ins Spiel. Man erzeugt Klassen, die aufeinander aufbauen. Die erste Klasse ist zum Beispiel vom Typ Treiber und beschreibt in ihrem Interface alles was ein beliebiger Treiber kennen muss.

Nun legt man für bestimmte Gerätegruppen Unterklassen an. Die Unterklasse (Subclass) erbt das Interface ihrer Superklasse(Superclass).

Die Subclass erbt auch die Implementation (den Programmcode) ihrer Superklasse, kann die aber durch eigens auf sie angepassten Code ersetzen.

Bei Treibern wären das z.B. Klassen für USB, Netzwerk, Grafik. Jeder dieser Treiber hätte wieder Unterklassen, bei den USB Klassen z.B. für verschieden Chipsätze unterschiedliche Treiber.

Hat man nun einen Computer mit einem macos unbekannten USB Chipsatz und einem passenden Kext dazu, so wird über den vorhin erwähnten Mechanismus eine Instanz der Klasse erzeugt, die macos nicht kennt. Aber da es sich um einen USB Chipsatz Treiber handelt, ist er eine Unterklasse der USB Klasse und davon kennt macos das Interface und kann also damit arbeiten.

Im IORegistryExplorer kann man die Superklassen einer Klasse sehen.

AppleACPIPCI

Class	AppleACPIPCI : IOPCIBridge : IOService : IORegistryEntry : OSObject

Die Klasse AppleACPIPCI basiert auf IOPCIBridge das auf IOService basiert usw.

Alle Treiber sind Unterklassen von IOService, das eine Unterklasse von IORegistryEntry, das eine Unterklasse von OSObject ist.

OSObject stellt Standardfunktionalität für Objekte im Kernel zur Verfügung. Das betrifft hauptsächlich die Speicherverwaltung, also wie man Objekte anlegt und löscht.

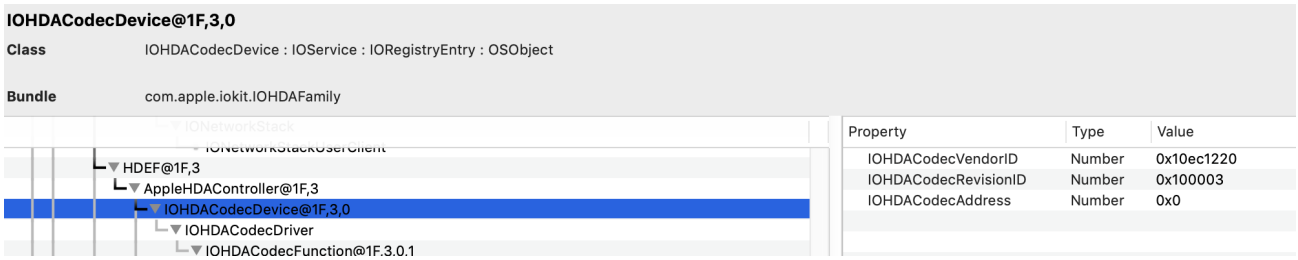
IORegistryEntry erweitert OSObject um Funktionen die ein Verwalten der Objekte in der IORegistry erlauben. Anlegen, verschieben und Löschen von Einträgen uä.

IOService fügt dann alles hinzu was jeder Service können muss. das ist ein A-voll Zeug wens interessiert, der kann's googeln.

Wir wissen das die Eigenschaft die wir ändern wollen IOHDACodecRevisionID heißt.

Diese kommt in IOHDACodecDevice vor.

Es besteht die Möglichkeit, dass die Eigenschaft von einem anderen Service übergeben wird. Aber das interessiert und erst, wenn das Ändern an der Stelle nicht funktioniert.



So an der Stelle haben wir die Instanz und die Klasse.

Die Klasse lautet ..... ?

Verwendet man IOProviderClass für das matching so wird man über jeden Service informiert, der der Klasse oder eine Unterklasse davon entspricht.

Gibt es mehrere Instanzen einer Klasse z.B. IOService, kann man die "Matches" weiter einschränken. Das ist dann der Schritt nachdem wir die Klasse bestimmt haben haben.

Ich verstehe die IORegistry Frage nicht.

HDEF@1F,3 und AppleHDAController@1F,3 sind unterschiedlicher Services.

Was meinst du mit unterschiedlichen Ansichten ? Service vs. ACPI ?