16.3.5: البنى

يتم تعريف البنى تماماً كما تعرف في لغة C++. لكن لا يمكن للبنى في لغة HLSL أن تمتلك مناهج. فيما يلي مثال عن بنية بلغة HLSL:

```
struct MyStruct
{
    matrix T;
    vector n;
    float f;
    int x;
    bool b;
};
MyStruct s;
s.f = 5.0f;
```

16.3.6: الكلمة المحجوزة typedef

إن وظيفة الكلمة المحجوزة typedef في لغة HLSL هي نفسها تماماً كما في لغة C++. مثلاً يمكننا إطلاق اسم point على النوع vector<float, 3> باستخدام الصيغة التالية:

```
typedef vector<float, 3> point;
```

عندئذ بدلاً من أن نكتب:

```
vector<float, 3> myPoint;
```

يمكننا فقط أن نكتب:

```
point myPoint;
```

هنا مثالين آخرين يبينان استخدام الكلمة المحجوزة typedef لتعريف نوع ثابت ونسق:

```
typedef const float CFLOAT;
typedef float point2[2];
```

16.3.7: سوابق المتحولات

يمكن استخدام الكلمات المحجوزة التالية كسابقة عند التصريح عن متحول:

- **static**: إذا كان متحول ما مسبوقاً بالكلمة المحجوزة static فهذا يعني بأنه لن يكون مرئياً خارج المظلل. بتعبير آخر سيكون هذا المتحول محلياً في المظلل.

  أما إذا تم إسباق متحول محلي بالكلمة المحجوزة static عندها سيسلك نفس سلوك متحول محلي static في لغة C++. أي يتم تهيئته مرة واحدة عندما يتم تنفيذ التابع