Structures

Just because you put your data in a struct doesn't mean your data is structured

Structured Data vs. Bag O' Data

Common pitfalls and easy tips.

Don't mix context information with data to be transformed.

These are two *unrelated* sets of data.

```
31  class InternalGridAttribute : public BaseInternalAttribute
32  {
33  public:
34    //--------------------
35    // Data
36
37    bool      m_Visible;
38
39    bool      m_Enabled;
40
41    //
42    // API
43
44    inline uint32_t GetGridSizeSetting( void ) const;
45    uint32_t        uint32_t Luna::InternalGridAttribute::GetGridSizeSetting(void) const
46
```

Example 1: How does the above relate to the below?

```
59      // Internal Data
60    protected:
61
62      uint32_t  m_GridSizeSetting;
63      Grid      m_LocalGrid;   //!< Draw the major and minor line grids
64
65      static const InternalAttributeFieldInfo m_AttributeFields[];
66    };
67
```

Example 2: Mixing state machine state and transformed data

```
40        {
41        public:
42          enum ModifiedState
43          {
44            kUnmodified,
45            kModified,
46            kModified_NeedResolve,
47            kModified_NeedDDLUpdate,
48          };

49          Handle                  m_Handle;

50
51
52          DDLInstance::MemoryCB*   m_InstanceMemoryCB;   // Pointer to the DDL instanc
53          DDLInstance::Instance*   m_Instance;           // Pointer to the DDL instanc
54          AttributeDef::Index      m_DefIndex;
55
56          AttributeDB*             m_DB;
57
58          inline bool IsInternalAttribute( void );
59
60          inline ModifiedState GetModifiedState( void );
61
62          //--------------------
63          // Internal Stuff
64        protected:
65
66          ModifiedState            m_ModifiedState;
67
68          BaseInternalAttribute*  m_InternalData;
69
70          friend class BaseAttributeAccess;
71
72          friend class AttributeDB;
73          inline Attribute* FreeListGetNext( void ) const;
74          inline void       FreeListSetNext( Attribute* next );
```

Example 3: Public, protected, private data

There's no such thing as "optional" data.

"Optional" is context information.
Separate that data.

Don't mix data with different lifespans.

"Lifespan

You'll spend

checking to se

Example:
Update classes.
Build-time data versus
runtime data.

For each vs. For all

"For each" considered harmful.

What if branching was one of the most expensive things you could do on the CPU? ...oh, wait.

Easy Step #1: SOA

SOA by default.

Combine as *appropriate*.

Easy Step #1: SOA

Combine based on use:

e.g. xyzw

iate.

Easy Step #1: SOA

Com

What about conflicting combinations?

This problem *already* exists.

Easy Step #1: SOA

Com

Wh

Can solve for cache coherency.

~2B PPU cycles/sec wasted on waiting for memory.

This p

And, of course... Concurrency.

Separating data by context makes it *much* simpler to solve for data concurrency.

Real-life example:
David Thall