AppHandleEvent chs if event was a form load event

```
if (event->eType == frmLoadEvent)
{
    // sets active form
    formID = event->data.frmLoad.formID;
    form = FrmInitForm(formID);
    FrmSetActiveForm(form);
    switch(formID)  // set callback for form
    {   case MainForm:
            FrmSetEventHandler(form,
                               MainFormHandleEvent);
    }
```

MainFormHandleEvent has a switch case depending on Form specific events

```
switch(event->eType)
{   case frmOpenEvent:
        form = FrmGetActiveForm();
        FrmDrawForm(form);
        handled = true
    default:
        break;
}
```

Overload more on event handling ⇒ ctlSelectEvent, menuEvent

Memory Management

- Since memory is at a premium on Palm devices memory management is usually done through handles:

A handle is essentially a ptr to a ptr

By using handles the OS can better manage its memory heap since it can move around blocks of memory than update *p and the program that has p can still find its data.

Basic 6 APIs:

To allocate:

```
MemHandle memory = MemHandleNew(50);
```

(allocates 50 bytes)

To free memory

```
MemHandleFree(memory);
```

To determine how much memory has been allocated

```
UInt32 size MemHandleSize(memory);
```

In order to read, write modify the memory one needs to lock it first. Then unlock when done. (Prevents OS from moving where we write while reading.

Ex)

```
FormType *form = FrmGetActiveForm();
FieldType *field;
MemHandle h;
Char *s;
field = FrmGetObjectPtr(form,
        FrmGetObjectIndex(form,
        MyNameField));
if(FldGetTextLength(field)>0)
{
  h = FldGetTextHandle(field);
  s = MemHandleLock((void*)h);
  FrmCustomAlert(AlertID, s, NULL, NULL);
```

Other Functions

MemSet

MemMove

MemHandleResize