Custom function » Developer

When working on a FileMaker solution, there are many situations where debugging information or feedback is required as part of the development process. This type of information is not suitable for display within the realm of normal application use.

Therefore, it is suggested that various portions of your solution utilize a Developer custom function. At its core, the function can be as simple as `Get (AccountPrivilegeSetName) = "[Full Access]"` and named according to the Custom function naming standards.

However, there may be environments where certain accounts are given access to this type of information. In such an event, watching for specific accounts may be desired. The following custom function will provide for this situation.

```plaintext
/**
 * =====================================================
 * Developer ( )
 * https://github.com/petrowsky/fmpstandards/blob/master/Functions/Developer.fmfn
 * PARAMETERS:
 * none
 * RETURNS:
 * (bool) True or False based on proper evaluation
 * DEPENDENCIES:
 * none
 * NOTES:
 * Because Filemaker Changed Get ( PrivilegeSetName )
 * Evaluate () must be used
 * RELEASE:
 * 2011-02-23
 * ==================================================================
 */

Let ( [ ~developers = List ( "" ; "" ); // Add names of the accounts which are valid developer accounts
        ~version = Get ( ApplicationVersion );
        ~isGo = PatternCount ( ~version; "Go" ) = True;
        ~versionNumber = GetAsNumber ( Substitute ( ~version ; "," ; "." ) ); // account for foreign versions using comma
        ~extendedPrivileges = // Add "developer" extended privilege to groups which are valid developer groups
                              If ( ~isGo or ~versionNumber  11;
                                  Evaluate ( "Get ( AccountExtendedPrivileges )" );
                                  /*else*/ Evaluate ( "Get ( ExtendedPrivileges )" )
                              )
    ];

    PatternCount ( ¶& ~developers &¶ ; ¶& Get ( AccountName ) &¶ )  1
    or PatternCount ( ¶& ~extendedPrivileges &¶ ; ¶& "developer" &¶ )  1
    or
    If ( ~isGo or ~versionNumber  11;
          Evaluate ( "Get ( AccountPrivilegeSetName )" ) = "[Full Access]";
          /*else*/ Evaluate ( "Get ( PrivilegeSetName )" ) = "[Full Access]"
    )
}
An example of this, within a script, would look like the following:

```plaintext
If [ Developer ]
  Show Custom Dialog ["Debugging info"; Debug ()]
End If
```

Note the use of the `Debug custom function`.