EINE 4GL DES 21. JAHRHUNDERTS

Von ASCII Terminals zu Smartphones

Genero®

In seiner Blütezeit hat IBM® Informix® 4GL die Geschäftssoftware dominiert, weil es genau zu diesem Zweck entwickelt worden war – zur kostengünstigen und schnellen Entwicklung von zuverlässigen und skalierenden Büroprogrammen.

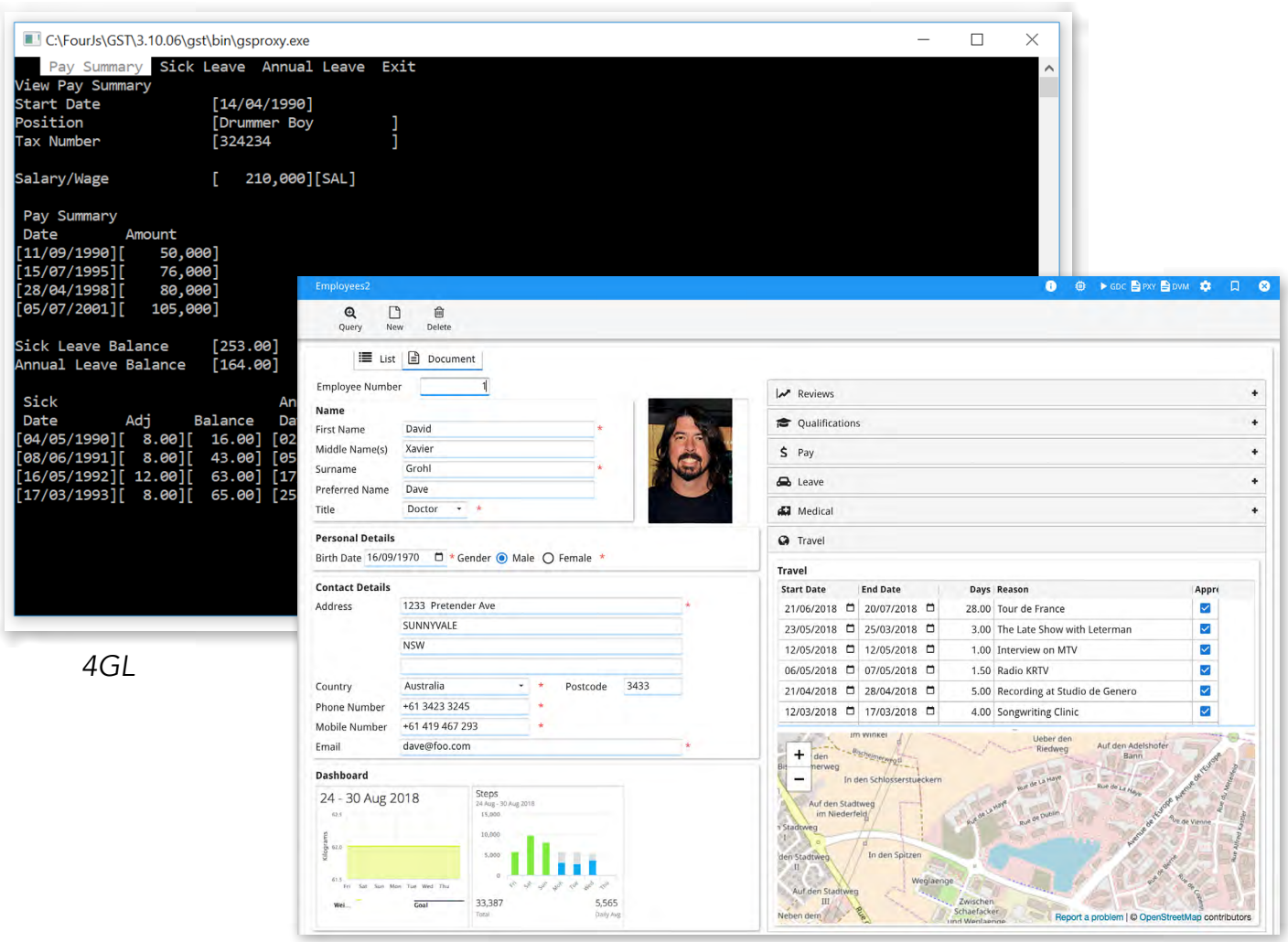
Mit dem Aufkommen von grafischen Benutzeroberflächen und objektorientierten Programmiersprachen in der Mitte der Neunziger Jahre verlor es jedoch seine Dominanz, auch wenn diese neuen Sprachen nicht die Produktivität von 4GL bieten konnten und können.

Dessen ungeachtet ist 4GL noch relevant und überall anzutreffen, in IT Landschaften mit knappen Budgets und limitierten Entwicklungszeiten. Es ist immer noch eine der produktivsten Arten, um Geschäftsanwendungen zu entwickeln.

In den letzten 30 Jahren haben die Technologiefortschritte unsere Arbeitswelt revolutioniert. Während dieser Zeit hatte 4GL aufgrund der fehlenden grafischen Oberfläche Schwierigkeiten mitzuhalten. Versuche seitens Informix, eine Weiterentwicklung mit GUI einzuführen, scheiterten mangels Kompatibilität mit dem Original 4GL. Deshalb wurde Genero® geschaffen. Genero ist eine komplette Neuimplementierung, die Anschluss an die neuesten Technologien bietet, aber dabei 100% mit dem Original codekompatibel ist. Genero sichert die beträchtlichen Investitionen in existierenden Quellcode und bewahrt auch die Erfahrung der Entwickler, die in den letzten Jahrzehnten angesammelt wurde. Es ist eines der portabelsten, erweiterbarsten und skalierbarsten Entwicklungssysteme, die es damit erstellten Applikationen ermöglicht, mühelos mit den aktuellsten Technologie-Trends mitzuhalten.

In seiner Genero Inkarnation laufen 4GL Programme überall – egal auf welchem grafischen Client, mit welcher Datenbank oder welchem Betriebssystem – mit keinen oder kleinen Änderungen. Sei es auf einem Terminal, Mac® oder MS® Windows Laptop, in einem Browser oder auf einem Smartphone, Programme adaptieren sich zur Laufzeit und müssen nur einmal entwickelt werden. Das erklärt, warum Genero weiterhin der Grund ist, 4GL zu benutzen und es immer noch das Rückgrat so vieler unternehmenskritischen IT Systeme von großen Firmen und staatlichen Einrichtungen ist.

> Genero schützt jahrzehntelange Investitionen in Code und Wissen



4GL

Genero

Das original 4GL hat sich nicht von seinen 80er 'Terminal Wurzeln' befreien können und nie ein Update für zeitgemäße Bedienung in einer grafischen Oberfläche bekommen. Deshalb sind auch heute Programme, die mit dem Original Produkt kompiliert werden, ausschließlich im Terminal bedienbar. Das heißt, es gibt weder eine bessere Bedienbarkeit unter Windows oder Mac, geschweige denn im Browser oder auf dem Smartphone.

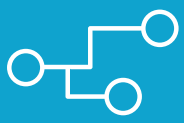
In Ermangelung von Registerkarten, Widgets und Symbolleisten neigen Anwendungen, die mit 4GL entwickelt wurden dazu, so viele Informationen wie möglich in ein einziges Formular zu komprimieren, was dann in «Master» - und «Detail» -Fenster aufgeteilt wird.

Der Einsatz von Genero klärt dieses mögliche Durcheinander, bringt die Benutzerbedienung auf den modernsten Stand und macht Anwendungen, ob alt oder neu, kaum noch unterscheidbar.

> Genero ist einmalig in seiner Fähigkeit, 25 Jahre alten Programmcode mit neuem Code zu kombinieren, der aktuelle Computerparadigmen bedient.

Zeitgemäße Ergonomie

In 4GL bieten Popup-Dialoge einen Mechanismus auf mehr Informationen zuzugreifen, jedoch muss vor einem neuen Aufruf der aktuelle Dialog beendet werden. Benutzer können von Feld zu Feld «springen», aber nur sequenziell und innerhalb eines Dialogs gleichzeitig. 4GL-Anwendungen sind nicht Browser-freundlich, verbunden mit dem Risiko Inhalte, durch Nutzung des «Zurück» - Buttons zu verlieren. Dies erscheint Nutzern, die mit modernen Systemen vertraut sind, als umständlich und letztendlich frustrierend. Mit hochmodernen Client-Rendering-Technologien löst Genero diese Probleme allesamt auf.

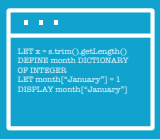


Interoperabilität

4GL ist, um es freundlich auszudrücken, introvertiert. Es kann nur mit der Datenbank interagieren. Moderne Applikationen – egal in welcher Sprache sie geschrieben wurden – interagieren untereinander mit Hilfe von SOAP und RESTful Webservices unter Benutzung von XML und JSON Datenstrukturen. Diese Möglichkeiten sind in Genero hinzugefügt worden, um Applikationen «gesprächiger» zu machen – endlich ist es möglich, in 4GL mit der Außenwelt über standardisierte Protokolle zu kommunizieren wie alle anderen modernen Programmiersysteme.

Sicherheit

Applikationen, die über das Internet bedienbar sind, werden von allen Seiten durch mögliche Attacken bedroht. Die Entwicklung von sicheren Programmen, die sich problemlos in komplexe IT Landschaften einfügen, sind deshalb missionskritisch. Genero stellt Klassenbibliotheken zur Verfügung, die Verschlüsselungsalgorithmen, Athentifizierungsmöglichkeiten, Zertifikatshandhabung und Single Sign-on bereitstellen, um Applikationen die sichere Interaktion mit Online Internetdiensten, sozialen Netzwerken und anderen Systemen zu ermöglichen.



Verbesserungen auf Sprachebene

Die Sprache hat weitreichende Erweiterungen erfahren, wie zum Beispiel die Möglichkeit, direkt Java Klassen und Funktionen aus 4GL heraus aufrufen zu können oder andere 4GL Module importieren zu können, um eine bessere Typprüfung bei der Übersetzung zu ermöglichen. Des Weiteren gibt es einen Satz von eingebauten Standardmodulen, welche die bequeme Interaktion mit dem Betriebssystem, dem Filesystem und TCP Servern ermöglichen. JSON Strukturen können auf einfachste Art auf 4GL RECORDS zugewiesen werden und umgekehrt. Es gibt einen neuen DICTIONARY Datentyp, die Möglichkeit, FUNCTION als Parameter zu übergeben und die Möglichkeit, generische Dialoge zu schreiben, Alles Dinge, von denen ein 'klassischer' 4GL Programmierer nur träumen konnte.

Eine integrierte Entwicklungsumgebung

Jede moderne Programmiersprache braucht eine umfangreiche Sammlung an grafischen Werkzeugen, um effizient zu entwickeln. Genero Studio, Genero Report Writer and Genero Ghost Client wurden für diesen Zweck geschaffen – die Produktivität der Entwickler zu verbessern, sowie Applikationen automatisch auf Stabilität, Zuverlässigkeit und Skalierbarkeit testen zu können.

Architektur

Genero befreit die Entwickler von den 4GL Fesseln – vor allem von seiner Mono-Datenbank, seinem Mono-Betriebssystem sowie seiner monolithischen und Greenscreen-Kultur. Durch die Befreiung von diesen Zwängen haucht Genero Altanwendungen neues Leben ein und bereitet sie für heutiges und morgiges Einsatz-Spektrum vor. Erreicht wird das mit einer einzigen Entwicklungsumgebung zum Erstellen von Anwendungen, die sicher, skalierbar und nahezu unbegrenzt portierbar ist, und von Tausenden von Nutzern ohne teure Middleware verwendet werden kann.

Zeitgemäße Ergonomie

Es ist möglich, Informix 4GL Applikationen mit Genero neu zu übersetzen und sie genau wie bisher im Textmodus (TUI) weiter zu benutzen. Allerdings ist das wenig sinnvoll, wenn man bedenkt, dass die gleiche Applikation nach der Übersetzung durch Genero automatisch und ohne weitere Änderungen am Quellcode (!) mit einer grafischen Oberfläche startbar ist, dem sogenannten „traditionellen Mode".

Der traditionelle Modus muss die Geometrie der ASCII Oberfläche des Terminals zwar noch 100% nachbilden, kann den Endanwender aber mit Hilfe einer Buttonleiste besser durch das Programm führen, in der alle gültigen Funktionstasten der Applikation angezeigt werden und auch über Mausbedienung erreichbar sind. Aktive Eingabefelder sind frei mit der Maus erreichbar. Auch ist in der GUI völlig klar, in welche Felder Werte eingetragen werden können und welche nur Informationen anzeigen.

Die größte angenehme Überraschung bietet Der traditionelle Modus für Informix 4GL Programmierer, wenn DISPLAY ARRAY oder INPUT ARRAY Anweisungen im Programm enthalten sind und dann in der GUI automatische Bildlaufleisten (scrollbars) neben den Listen auftauchen (sofern die Liste mehr Einträge hatte als angezeigt). Es wäre also wenig sinnvoll, den GUI Modus von Genero nicht zu benutzen.

Der traditionelle Modus ist die schnellste Form des Umstiegs auf eine grafische Benutzeroberfläche, aber aufgrund der Nachbildung der ASCII Terminal Geometrie sehen die Applikationen trotzdem noch nicht zeitgemäß aus. (Ungeachtet dessen gibt es immer noch Kunden, die den traditionellen Mode aufgrund seiner Einfachheit und Wartbarkeit nutzen und schätzen). Für den zeitgemäßen Oberflächen Look gibt es in Genero den «normalen» GUI Mode. Genero Programmierer benutzen diesen Mode, indem sie in den altbekannten .per Dateien eine LAYOUT Sektion anstelle einer SCREEN Sektion anlegen. Dieses Schlüsselwort ermöglicht die viel freiere Platzierung von Eingabeelementen auf der Eingabemaske und führt neue Elemente ein, die typisch für Programme mit modernem GUI Design sind.

In vielen Fällen ist es bereits ausreichend, nur durch die Modifikation der .per Dateien eine hinreichend moderne Oberfläche zu präsentieren. Genero erweitert die .per Syntax um «abstrakte» grafische Oberflächenelemente, die auf den verschiedenen Endanwender Systemen wie Windows, Linux®, MacOS, Browser und Smartphones dann «nativ» geändert werden: So sieht z.B. eine «Checkbox» unter Windows aus wie native Kontrollkästchen je nach Windows Inkarnation und unter IOS wie der dort bekannte Schiebeschalter.

Der Clou an Genero ist, dass, egal wie Kontrollkästchen gerade aussehen (ob mit oder ohne Rand, ob sie beim Ein/Ausschalten das Häkchen beeindruckend animieren oder nicht) sich die Programmierschnittstelle in 4GL dafür nicht ändert.

Seit der Einführung von Informix 4GL kann man mit der DISPLAY Anweisung Werte auf Eingabefelder zuweisen und mit der INPUT Anweisung Werte von Eingabefeldern einlesen. Auch nach der Einführung von neuen graphischen Bedienelementen hat sich das, aufgrund der Stärken von Genero, nicht geändert.

Graphische Bedienelemente definieren , wie Werte in einer Maske eingegeben und angezeigt werden:

- BUTTON – präsentiert eine klickbare/ antippbare Schaltfläche, um eine Aktion im Programm auszulösen
- BUTTONEDIT – ein Eingabefeld mit integriertem «Button», um kontextspezifische Ereignisse für das Eingabefeld auslösen zu können
- COMBOBOX – Endanwender können Werte in einer ausklappbaren Liste auswählen
- DATEEDIT – ein Eingabefeld spezifisch zur Datumseingabe, Endanwender können sowohl mit der Tastatur das Datum eingeben, oder aber durch Anklicken/Antippen des integrierten Buttons das Datum über einen Dialog auswählen
- DATETIMEEDIT – ein Eingabefeld, um DATETIME Variablen bequemer eingeben zu können
- EDIT – das Standard Eingabefeld, um Werte einzugeben und anzuzeigen
- IMAGE – stellt Bilder dar
- LABEL – stellt nicht editierbare Texte dar, z.B. Feldnamen oder Zusatzinformationen
- PROGRESSBAR - ermöglicht es, Verläufe als Fortschrittsbalken zu visualisieren
- RADIOGROUP – ermöglicht es, einen von n Werten zu selektieren
- SLIDER – Schieberegler, ermöglicht die grafische Auswahl eines Wertes
- SPINEDIT – Endbenutzer können einen ganzzahligen Wert über die Tastatur eingeben und zusätzlich über 2 Buttons den Wert erhöhen und verringern
- TEXTEDIT – ermöglicht die mehrzeilige Texteingabe
- TIMEEDIT - Benutzer können die Zeit bequem über dieses spezialisierte Element eingeben

Layout Manager legen fest, wie Bedienelemente auf der Oberfläche in Relation zu anderen Bedienelementen, positioniert werden.

- GRID – positioniert einzelne Unterelemente nicht überlappend und ausgerichtet in Zeilen und Spalten mit Abständen und ist eine direkte Weiterentwicklung der SCREEN Sektion
- SCROLLGRID – arrangiert Mehrfach Anordnungen von GRID Elementen
- TABLE – sorgt für die tabellarische Darstellung von Datenbankspalten
- TREE – ähnlich wie TABLE, aber mit einer linksseitigen zusammenklappbaren hierarchischen Baumstruktur
- STACK – Subelemente werden adaptiv angeordnet, so wie es das jeweilige Endanwender System vorschlägt
- GROUP – gruppiert Unterelemente und platziert einen Titel darüber
- FOLDER + PAGE – ermöglich die typische Karteireiter Ansicht mit jeweils einer sichtbaren Karteikarte, die anderen erscheinen als Tabs oder faltbare Akkordeons
- VBOX – stapelt Unterelemente vertikal
- HBOX – stapelt Unterelemente horizontal

Webkomponenten – ermöglichen die Benutzung von Drittanbieter Komponenten, die mit HTML, CSS und JavaScript erstellt wurden sowie die Entwicklung von kundenspezifischen Bedienelementen in JavaScript/HTML. Beispiele dafür sind Kalender, Diagramme, Bedienelemente zur Stift-Eingabe (Unterschrift), Texteditoren, Bild- und Videobetrachter, Kassensysteme usw.

Vordefinierte Webkomponenten - stellen einen erweiterten Texteditor, einen Bildauswahldialog und ein SVG Programmierinterface bereit.

Action defaults - zentralisieren die Eigenschaft für die ON ACTION Anweisungen im Programm und ermöglichen es, für mehrere ON ACTION Anweisungen mit demselben Aktionsnamen die Vergabe eines Textes, Kommentars, Bildes und Tastaturkürzel an einer Stelle zu spezifizieren.

Top menus (Menüleiste) – bietet eine Definition für Standard Menüleisten, deren Einträge bei Auswahl ON ACTION Anweisungen auslösen werden.

Toolbars (Symbolleiste) – bietet die Definition einer Standard Symbolleiste, die an allen 4 Seiten eines Formulars angeordnet werden kann. Bei Auswahl eines Eintrags werden ebenfalls ON ACTION Anweisungen im Programm ausgelöst.

Parallele Dialoge – vor parallelen Dialogen konnte bisher ein Fenster (WINDOW) nur per Programmcode aktiviert werden (über CURRENT WINDOW IS). Jetzt können Endbenutzer unter mehreren Fenstern frei auswählen, sofern die Programmlogik dafür adaptiert wurde.

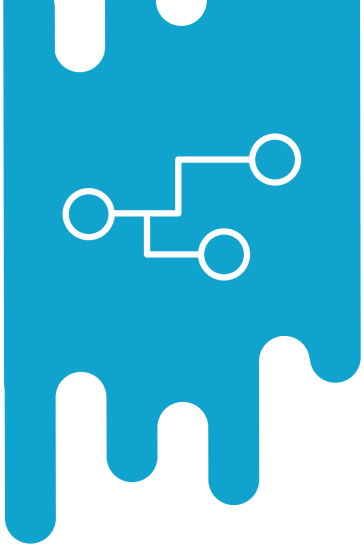
Multiple Dialoge – können mehrere traditionelle Anweisungen für die Interaktion (INPUT, INPUT ARRAY, DISPLAY ARRAY) gleichzeitig ablaufen lassen. Z.B. kann man gleichzeitig mit DISPLAY ARRAY in einer Liste scrollen und Werte über einen INPUT eingeben, ohne dass die DISPLAY ARRAY Anweisung oder der INPUT programmtechnisch verlassen werden muss.

Dynamische Dialoge – können Interaktionsobjekte, die Anweisungen für die Interaktion zur Kompilierzeit entsprechen (INPUT, CONSTRUCT, DISPLAY ARRAY, INPUT ARRAY, DIALOG,…) zur Laufzeit instanziieren. Das erlaubt das Erstellen von generischem Code mit variablen Datenstrukturen (ohne festes Datenbankschema). Zusammen mit der Möglichkeit, Eingabemasken zur Laufzeit zu generieren und dynamische Datenbankanweisungen mit der base.Sql-handle Klasse ermöglicht dies große Einsparungen von Programmcode in Bereichen, in denen man bisher ständig wiederkehrenden Boilerplate Code schreiben oder generieren musste.

Multiple Selektion – die Möglichkeit, in einem DISPLAY ARRAY nicht nur die aktuelle Zeile zu markieren, sondern auch mehrere Zeilen gleichzeitig zu selektieren und im Programmcode abzufragen, sowie die selektierten Zeilen per Drag und Drop an andere 4GL und Desktop Programme weiterzugeben.

Focus one field (Fokussiere eine Zelle) – Schaltet die Standard Hervorhebung einer Tabellenzeile auf die Selektion einer einzelnen Zelle um.

Drag and drop – ermöglicht das Herausziehen von Tabellenzeilen oder Feldinhalten mit der Maus entweder auf andere Tabellen/Eingabefelder im aktuellen Programm oder auch auf andere Programme.



Interoperabilität

Genero hat sich in den letzten 25 Jahren weiterentwickelt, um die aktuellsten Programmierschnittstellen und Datenstrukturen bereitzustellen, die heutzutage benutzt werden und, um mit anderen Systemen zu interagieren. Das beinhaltet die wohl populärsten Datenstrukturen XML und JSON zusammen mit der Möglichkeit Webservices aufzurufen, sowohl via SOAP als auch über RESTful .

Daten Structures

XML

Genero hat eine umfangreiche Anzahl an Klassen und Methoden, um mit XML Dokumenten zu hantieren. Sie beinhalten Unterstützung sowohl für das Document Object Modell (DOM), als auch für die Streaming Programmierschnittstelle StAX. Weiterhin werden Serialisierung und XSLT Transformationen unterstützt.

JSON

Genero stellt auch Klassen bereit, um via JSON Format (JavaScript Object Notation) Daten mit anderen Programmen austauschen zu können. Dabei ist der Weg über JSON extrem einfach, mit nur einer Zeile Programmcode kann man einen 4GL RECORD in eine JSON Zeichenkette umwandeln und umgekehrt.

Methoden

Webservices – Genero stellt Bibliotheken zur Verfügung, um auf einfache Weise Webservices zu erstellen (Server Teil in 4GL) oder den Zugriff auf existierende Webservices zu ermöglichen (Klienten Teil in 4GL).

SOAP – ein sehr bekanntes Verfahren, in dem eine WSDL Datei die Aufruf Parameter definiert. Über ein Werkzeug namens fglwsdl kann der Server- oder Klienten-Programmcode in 4GL generiert werden.

RESTful – die leichtgewichtige Alternative zu SOAP. Überträgt Daten (typischerweise JSON oder XML) mit HTTP und nutzt die HTTP Infrastruktur, um die Übertragung effizienter zu gestalten.

File – Stellt eine Schnittstelle zum Host Dateisystem dar und eliminiert deshalb die vorher gebräuchlichen C-Erweiterungen, um Dateizugriffe zu ermöglichen.

Base.Channel – informiert 4GL Programmierschnittstelle, um Dateifunktionen aufzurufen. Diese war aber so dürftig, dass die meisten Firmen eigene C Erweiterungen und Shell Skripte pflegen mussten, um diese Funktionalität zu nutzen. Die base.Channel Klasse bietet eine Anzahl von Funktionen, um Dateien zu lesen und zu schreiben und kann existierende C Erweiterungen und Shell Skripte vollständig ersetzen (um z.B. eine Portierung auf ein anderes System zu erleichtern). Natürlich werden C Erweiterungen immer noch unterstützt.

OS – Diese Klasse ermöglicht portable Operationen auf Pfaden und Dateinamen (wie zum Beispiel die Ausgabe des aktuellen Dateipfades). Dies ermöglicht es, einen vom Betriebssystem unabhängigen Programmcode zu erstellen.

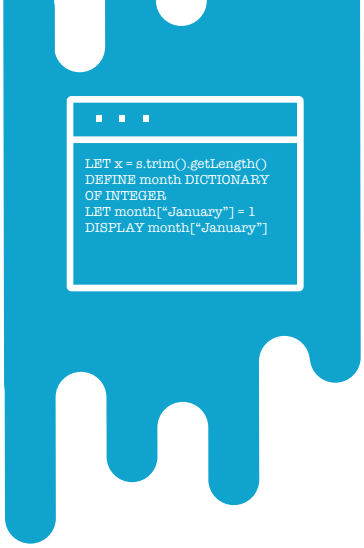
Sockets – Methoden sind gleichfalls in der base.Channel Klasse enthalten, um über TCP Sockets entweder einen Socket Client zu implementieren oder auch einen TCP Server bereitzustellen.

Pipes – des Weiteren wurde die open-Pipe Methode zur base.Channel Klasse hinzugefügt, um mit Subprozessen über eine Pipeline wahlweise unidirektional oder bidirektional zu kommunizieren.

Sicherheit

Security – Diese Bibliothek stellt Klassen und Methoden zur Kryptographie zur Verfügung. Sie beinhaltet PBKDF2 und BCrypt Klassen für die Speicherung und Verifikation verschlüsselter Kennwörter. Weitere Klassen implementieren bekannte benötigte Algorithmen wie z.B. SHA512, MD5 usw. sowie die Konvertierung in Binär-,Hexadezimal-undBase64-Formate.

Sichere Kommunikation – Wenn eine Applikation Nachrichten mit anderen Applikationen im Internet austauschen möchte, (sei es eine Finanz-, Büro- oder andere Applikation) muss sie in der Lage sein, die Authentizität der Nachricht zu gewährleisten und durch Verschlüsselung sicherzustellen, dass keine bösartige Drittapplikation die Nachricht entweder dekodieren oder sogar verändern kann. Das alles wird durch den zeitgemäßen Gebrauch von Zertifikaten, kryptografischen Schlüsseln und die Ende zu Ende Verschlüsselung von Genero sichergestellt.



Sprachverbesserungen

Die 4GL Sprachsyntax wurde weitgehend erweitert, um einige Limitierungen zu beseitigen und gleichzeitig die Sprache grundlegend zu modernisieren. Hier eine Liste einiger neu hinzugekommener Schlüsselwörter:

Syntax

Schlüsselwörter – STRING, CONSTANT, BOOLEAN, TYPE, BIGINT, TINYINT, PUBLIC, PRIVATE, NVL, IFF, SFMT, DICTIONARY, DYNAMIC ARRAY, TRY/CATCH

ON ACTION – Die ON ACTION Anweisung führt Programmcode aus, wenn ein externes Ereignis eintritt:

```
ON ACTION zoom
```

print lässt den Zweck der Programmzeile besser erkennen als:

```
ON KEY(F5)
```

Die Möglichkeiten, mit denen die 'print' Action ausgelöst werden kann, können entweder in einer .per Form Datei spezifiziert werden oder in einer «Action default « Datei. Ob die Aktion am Ende über einen Mausklick oder ein Tastaturkürzel ausgelöst wurde, ist egal. Im Vergleich zu den traditionellen ON KEY Anweisungen ermöglicht dies ein höheres Abstraktionsniveau in den Dialogen und erhöht die Wartbarkeit entsprechender Programmzeilen.

Funktionsaufrufe mit Referenzwerten – Diese Eigenschaft ist nützlich, wenn komplexe und grosse Feldstrukturen an Unterroutinen übergeben werden. Es ist möglich, diese Strukturen als Referenz zu übergeben. Somit existiert nur eine Instanz des Feldes im Speicher und verhindert zeitintensive Duplikation der Daten nur zum Zweck des Aufrufes einer Unterroutine (und ermöglicht somit den Verzicht auf modulglobale Felder).

```
DEFINE arr DYNAMIC ARRAY OF complexDataType

# arr contains untransformed data
CALL transform_array(arr)
# arr contains transformed data
```

Literal Initialisierung – Variable können einfach mit Literalen initialisiert werden:

```
TYPE colorType RECORD
   red, green, blue INTEGER
END RECORD

DEFINE purple colorType = (255,0,255)
```

Case sensitivity (optional) - Der Compiler kann angewiesen werden dies als ungültig zu betrachten:

```
DEFINE foo, Foo, FOO STRING
```

und:

```
DEFINE accountCode STRING
DEFINE accountCodeLength INTEGER
        LET accountCodeLength = accountCode.getlength()
```

Namen für Funktionsparameter beim Aufruf (geplant) – Namen für Funktionsparameter erleichtern wesentlich die Lesbarkeit und damit die Wartbarkeit für Funktionsaufrufe. Die Reihenfolge der Parameter muss dann nicht mehr zwingend eingehalten werden.

```
CALL fgl_report_configurePDFDevice(fromPage = 1, toPage = 1)
```

mögliche Typdeklaration – in der Parameterliste, Methoden Verkettung, Hashtabellen

Typdeklarationen im Funktionskopf erlauben dem Compiler eine strengere Typprüfung und sparen Schreibarbeit.

```
FUNCTION add(x INTEGER, y INTEGER) RETURNS INTEGER
```

Methoden Verkettung ist üblich in anderen Sprachen, war vor Genero nicht möglich.

```
LET x = s.trim().getLength()
```

Hashtabellen erweitern Genero um einen neuen Datentyp mit einer Zeichenkette als Schlüssel, um auf den Wert in der Hashtabelle zuzugreifen.

```
DEFINE month DICTIONARY OF INTEGER
    LET month["January"] = 1
    DISPLAY month["January"]
```

***ANYRECORD** – Generischer RECORD Typ*
Der ANYRECORD Typ erlaubt es, generische Funktionen zu erstellen, die mit verschiedenen konkreten RECORD Instanzen aufgerufen werden können. ANYRECORD beinhaltet auch Methoden, um über die einzelnen Felder und Sub Records eines RECORDs zu iterieren und Namen, Typen und Werte abzufragen bzw. zu setzen.

IMPORT – Wird benutzt, um externe Bibliotheken oder Genero Module einzubinden. Im Einzelnen gibt es:

- `IMPORT FGL` - Import eines Genero 4GL Moduls
- `IMPORT` - Import einer 'C' Bibliothek
- `IMPORT JAVA` - Import einer Java Bibliothek

Unsere Entwicklergemeinde benutzt IMPORT JAVA u.a. für folgende Zwecke:
- Erstellung von MS® Word und Excel Dateien unter Benutzung der Apache POI Bibliothek
- Bearbeitung von pdf Dokumenten über verschiedene Java Bibliotheken
- Versenden von Mail via javax.mail

Internationalisierung – Genero Applikationen können überall auf der Welt verwendet werden. Es werden die traditionellen Zeichensätze unterstützt, die auch Informix 4GL bietet (GLS), zusätzlich wird auch UTF-8 unterstützt. Neu ist die Möglichkeit, Zeichenketten zeichenweise zu indexieren, im Gegensatz zur bisherigen Indexierung über einzelne Bytes (Umgebungsvariable FGL_LENGTH_SEMANTICS=CHAR). Gerade bei Sprachen, die nicht im europäischen Raum gesprochen werden, oder beim UTF-8 Zeichensatz gilt die Regel «ein Byte für ein Zeichen» nicht mehr. Das Problem tritt bei substring Operationen auf und bei Ermitteln des nten Zeichens in einer Zeichenkette (String.getCharAt).

Eine *lokalisierte Zeichenkette* ist Text, der in einer Applikation benutzt wird und sich in sprachspezifischen Übersetzungsdateien außerhalb des Programmcodes befindet. Auf diese Weise kann für Programme die Sprache für Endbenutzer vor dem Start als Konfiguration gesetzt werden, ohne dass das Programm neu übersetzt werden muss.

Eine *Applikations Lokale* legt fest, für welche Region welche Übersetzungsdatei benutzt werden soll abhängig vom benutzten grafischen Client. Auf diese Weise kann die gleiche Applikation Clients mit verschiedenen Sprachen gleichzeitig bedienen. Dabei wird auch Rechts nach Links Darstellung (Arabisch, Hebräisch) unterstützt.

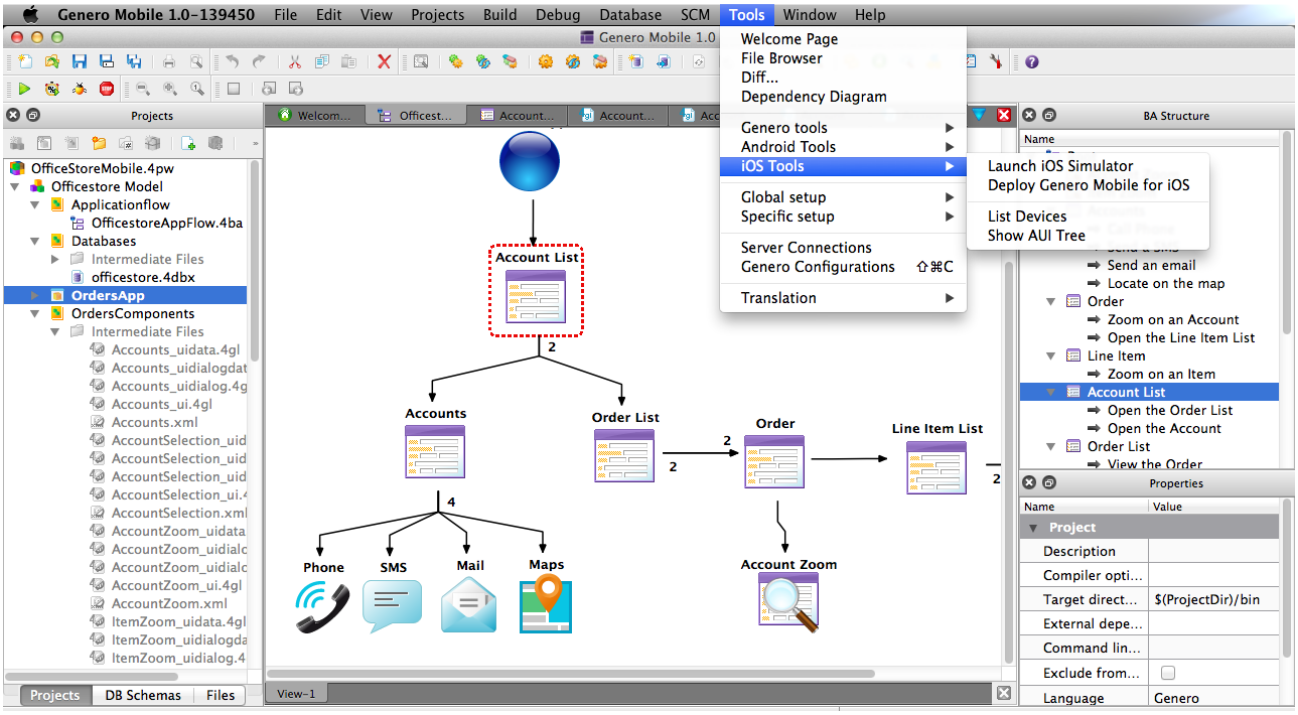
Eine integrierte Entwicklungsumgebung

Genero Studio ist eine integrierte Entwicklungsumgebung (IDE), die geschaffen wurde um die Produktivität der Entwickler zu steigern. Sie ermöglicht die Entwicklung und die Auslieferung von Programmen, die im großen Maße Betriebssystem agnostisch sind. Genero Programme können u.a. auf den folgenden Systemen ohne Neukompilation laufen: MS Windows, MacOS, Linux, Unix®, iOS®, Android®.

Studio enthält einen sprach-sensitiven Code Editor, der Code Komplettierung, permanenten Syntax Check, Einfärben von Schlüsselwörtern von 4GL und auch andere Dateiformate unterstützt. Weiterhin beinhaltet es einen grafischen Debugger, einen grafischen Masken Designer, einen grafischen Editor für Rechnungsdrucke, einen Projektmanager für komplexe Applikationen, einen Datenbankschema/Metaschema Generator, ein visuelles Modellierungs-Werkzeug sowie Profiler- und Analyse-Werkzeuge.

Leistungsfähige Entwicklungswerkzeuge

Genero Studio Business Anwendungs- Modeller (BAM) ist eine Low-Code-Entwicklungsplattform (LCDP), ideal geeignet für Entwickler, die auf der Suche nach einer inkrementellen, kontinuierlich erweiterbaren Cross-Plattform-Entwicklung sind. Der komplette Lebenszyklus von Entwicklungen wird mithilfe von Diagrammen und nicht mit Hand-Codierung erreicht.



BAM - ermöglicht die Integration nativer Smartphone-Apps in ein Genero Mobile-Modul.

Code für Applikationen und Webservices ist in Vorlagen definiert und visuell modelliert, was sowohl Konsistenz als auch strukturierten Code garantiert, der wiederum in Schichten organisiert ist.

- Speicherung von Daten
- Datendienste
- Web-Dienste Publikation (RESTful, SOAP)
- Formulare und Berichte

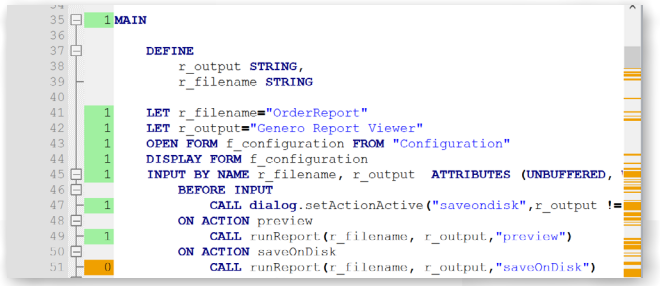
Die meisten businessspezifischen Codierungsmuster fallen in Standard-Vorlagen als «CRUD» Formulare, Listen, Details, Filterung, Navigation, Parallelitätsverwaltung und Berichte an. Native mobile apps, z. B. Kontakte, Kalender und Telefon sind mittels Beschreibung in das Modell komplett integrierbar. Dadurch können große Teile der Anwendung als Diagramme ohne eine Codierung überhaupt beschrieben werden. Diagramme werden im Laufe der Zeit durch Entwickler geändert, mit dem Ziel, sie zu verbessern oder Anpassungen an neue Bedingungen und neues Nutzerverhalten vorzunehmen. Maßgeschneiderte Business-Regeln können in sogenannten „Events“ kodiert werden.

Leistungsstarke Entwicklungsunterstützung

Diese Tools können während Kompilierung oder Laufzeit von Genero Studio oder von der Befehlszeile aktiviert werden.

Profiler – Profiler identifiziert Programm-Engpässe. Er zeigt den Nutzungsgrad von Funktionen in Prozent und wie häufig diese von anderen Funktionen aufgerufen werden.

Code coverage – Das Quellcode-Coverage-Tool analysiert, ob ein Testprozess jede Codezeile in einer Anwendung testet. Wenn sie ein Programm mehrmals mit aktiviertem Code Coverage ausführen, wird für jede einzelne Codezeile, die ausgeführt wird, ein Zähler beibehalten.



Trace – Trace ist eine nützliche Debugging-Funktion, die ein vollständiges oder gefiltertes Protokoll von Funktionsaufrufen anfertigt, die von einem Programm erstellt wurden, welches die übergebenen Argumente waren und was zurückgegeben wurde.

Shared libraries – Die Deployment-Architektur basiert auf Shared Libraries, wodurch keine «Runner» mehr erstellt werden müssen.

Werkzeuge zum Einrücken und Refactoring

Automatisiertes Testen (Genero Ghost Client)

Der Genero Ghost Client kann:
- unit Tests ausführen, um Abläufe in der Geschäftslogik zu validieren
- load Tests ausführen, um zu überprüfen, wie viele gleichzeitige Benutzer eine Infrastruktur unterstützen kann
- Messung von Antwortzeiten unter hoher Last

Ein graphisches Reporting Tool (Genero Report Writer)

Klassische 4GL Applikationen produzieren Berichte mit Hilfe der REPORT Anweisung. Die REPORT Anweisung bietet die Möglichkeit, einen Bericht mit fixer Zeichenbreite auf dem Bildschirm anzuzeigen, ihn in eine Textdatei zu schreiben, oder auf dem Drucker auszugeben. Mit heutigen Standards gemessen sehen diese Berichte nicht mehr zeitgemäß aus.

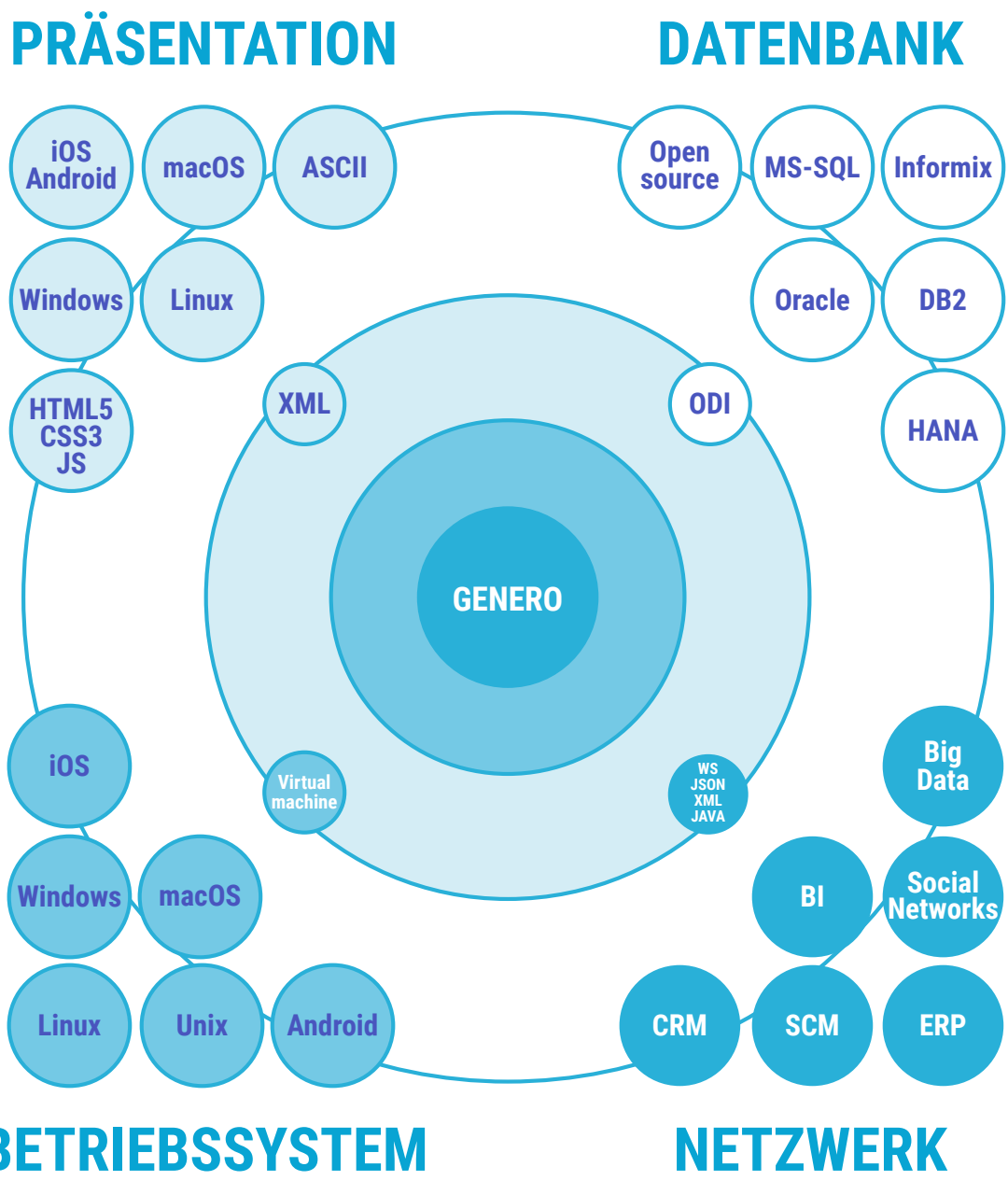
Genero Report Writer führt grafische Berichte ein, die viele verschiedene bekannte Ausgabeformate unterstützen (PDF, SVG, XSLX, DOCX, HTML, Postscript, JPG/PNG) und natürlich sowohl auf dem Bildschirm in einer Datei als auch auf dem Drucker ausgegeben werden können. Diese grafischen Berichte können Bilder, Tabellen, Strichcodes, Diagramme, Grafiken und eine Vielzahl von proportionalen Schriftarten in allen möglichen Farben und Stilen enthalten. Als Besonderheit bietet der Genero Report Writer eine «Streaming» Ausgabe an, die bei Übertragung eines Dokumentes auf die Benutzerseite keine unnötig großen temporären Dateien auf der Seite des Benutzers anlegen muss, nur damit die erste Seite angezeigt werden kann (Kurzes Beispiel: ein 1 GByte großer Bericht mit 1000 Seiten würde selbst bei einer schnellen Internetverbindung ziemlich viel Zeit in Anspruch nehmen, um komplett übertragen zu werden).

Berichte werden mit dem grafischen «Report Designer» erstellt, ein Werkzeug, das eine gute visuelle Kontrolle über das finale Seitenlayout ermöglicht und unnötige Testausdrucke überflüssig macht.

Übrigens kann «Genero Report Writer» auch aus anderen Programmiersprachen bedient werden wie z.B. aus Java, C und PHP.

Kompatibilitäts Modus – Der Kompatibilitäts Modus ermöglicht es, bestehende 4GL «Reports» grafisch ansprechender auszugeben und alle unterstützten Ausgabe Formate von «Genero Report Writer» zu zeigen.

Architektur



Sicher, portabel und skalierbar, für Tausende von Benutzern, ohne die Notwendigkeit teurer Middleware.

BÜROS

USA & Kanada
Four Js Development Tools Inc.

251 O'Connor Ridge Blvd. Suite 125
Irving, Texas, 75038
United States

Toll free Nummer : 866 3147300 (kostenlos)
Telefone : (972) 893-7300
Fax : (972) 893-7304

Latein Amerika
Four Js Development Tools Latin America SACV.

Insurgentes Sur 1602
Col. Credito Constructor
Mexico City, DF 03940
Mexiko

Telefone : +52 (55) 1000 9160

Europa
Four Js Development Tools Europe Ltd.

Suite 5, Rineanna House
Shannon Free Zone
Shannon V14 CA36
Co Clare, Irland

Telefone : +353 61 708 314
Fax : +353 61 708 315

Asien
Four J's Development Tools Asia

Suite 210 29 Kiora Rd,
Miranda NSW, 2228 Sydney
Australien

Telefone : +612-8004-5890

Frankreich
Four Js Development Tools SARL.

28 Quai Gallieni
92 150 Suresnes
Frankreich

Telefone : +33 1 41 38 86 30
Fax : +33 1 41 38 84 46

Großbritannien
Four Js Development Tools UK LTD.

Regus House, Victory Way Admirals Park
Dartford, DA2 6QD
Großbritannien

Telefone : +44 (0) 370 111 5140
Fax : +44 (0) 370 111 5154

Zentraleuropa
Four Js Development Tools Software Vertriebs GmbH

Laufzorner Weg 4
82064 Strasslach
Deutschland

Telefone : +49 8170 99 87 890
Fax : +49 8170 99 87 892

Iberica
Four Js Development Tools Iberica LTDA.

Martinez Villergas 49, 1a Planta,
28027 Madrid
Spanien

Telefone : +34 91 047 76 61
Fax : +34 91 047 76 57

Italien
Four Js Development Tools Italia

Via Cipriani, 2
42124 Reggio Emilia
Italien

Telefone : +39 (0)522 420786
Fax : +39 (0)522 420768

www.4js.com

FOUR Js
The Power of Simplicity