# iNEXT Suite 2013

(R)evolutionäre Zukunftsperspektiven für IBM i basierte Softwarelösungen


ML-Software GmbH

# Inhaltsverzeichnis

# Was macht iNEXT Suite 2013 so besonders?

Die neue iNEXT Suite 2013 wurde um die innovativen Technologien iNEXT Face und iNEXT ORM erweitert. Mit iNEXT Face werden die Displayfiles von Anwendungen auf AS/400 bis System i in die Windows-Welt migriert und stehen dort für die Weiterentwicklung und Programmierung zur Verfügung. Mit iNEXT ORM werden Objekte in .NET generiert, die den komfortablen Zugriff auf PF (physical File), LF (logical File) und SQL Abfragen optimal gewährleisten.

Die mit iNEXT Face migrierten Bildschirme unterstützen bereits alle vorhandenen Funktionen der IBM i Anwendungen, sind sofort einsetzbar und erhöhen die Benutzerfreundlichkeit spürbar. Sie können - müssen aber nicht - **im komfortablen .NET Designer optimiert** werden. Schon geringe Anpassungen wie neue Felder für die Anzeige von Dokumenten, Bildern etc. bringen Zusatznutzen. Die weitreichenden Möglichkeiten von .NET und Visual Studio können nach Belieben ausgeschöpft werden, um **Aufbau, Inhalt und Design aller Masken optimal an die Bedürfnisse der Nutzer anzupassen**.

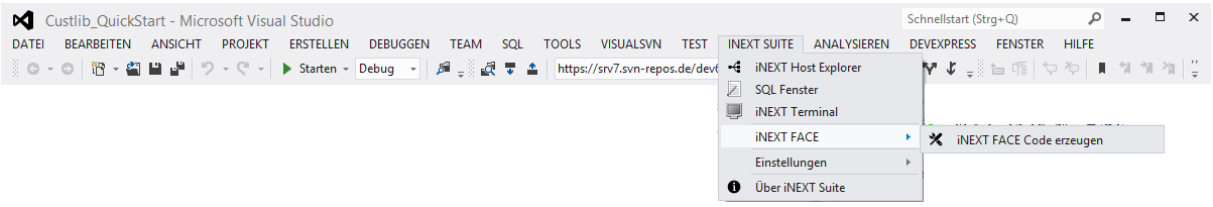
Die konsequente **Trennung von Businesslogik und Benutzeroberflächen** bildet die Basis für die **perfekte Zusammenführung von IBM- und Windows-Lösung**. Sie stellt sicher, dass **keine Änderungen an den RPG Programmen** notwendig sind. Und mehr noch: die iNEXT 2013 Screens können beibehalten werden, auch wenn die Businesslogik darunter verändert oder ausgetauscht wird. So können bei Bedarf ganze RPG-Programme oder einzelne Programmteile z.B. durch Neuentwicklungen in modernen .NET-Programmiersprachen ersetzt werden, was **Flexibilität und Zukunftssicherheit** in der IT bedeutet.

Durch diese Erweiterung deckt iNEXT Suite 2013 alle Facetten der Softwareentwicklung und -modernisierung ab. Das fängt an beim schnellen und risikofreien GUI on-the-fly, geht über die effiziente, regelbasierte und programmierbare Guisierung bis hin zur optimierten Bildschirmmigration mit iNEXT Face.

**iNEXT Suite 2013 bietet damit flexible und hoch effiziente Lösungen für alle IT-Anforderungen.**

# Wie funktioniert iNEXT Suite 2013?

iNEXT Suite 2013 nutzt als Entwicklungsumgebung das Visual Studio von Microsoft. Es wird durch die Installation voll im Hauptmenü integriert und bietet Ihnen so für Ihre effiziente Arbeit das gesamte Leistungsspektrum des Visual Studios wie z.B. komfortable Designer, leistungsfähige Komponenten, verschiedene .NET-Programmiersprachen, umfangreiche Programmierhilfen und vieles mehr.

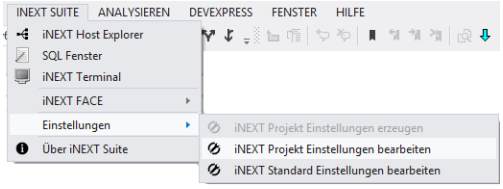



Zusammen mit iNEXT Suite 2013 wird auch ein Basisprojekt ausgeliefert. Es enthält ein lauffähiges Grundgerüst mit diversen Funktionalitäten wie z.B. Farb-/Font-Einstellungen, verschiedene Ansichtsmodi, Internetlinks und erleichtert Ihnen so den Einstieg in die Arbeit mit iNEXT Suite.

# Schrittweise Entwicklung eines iNEXT 2013 Clients

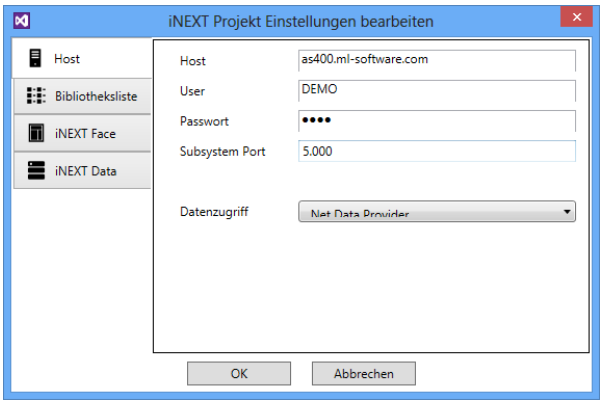
## Schritt 1: GUI on-the-fly in 5 Minuten

Sie laden das Basisprojekt und passen dann die Einstellungen an Ihre IT-Umgebung an. Öffnen Sie dazu im Menüeintrag „iNEXT Suite“ die „Einstellungen“ und klicken Sie dann auf den Eintrag „iNEXT Projekt Einstellungen bearbeiten“.

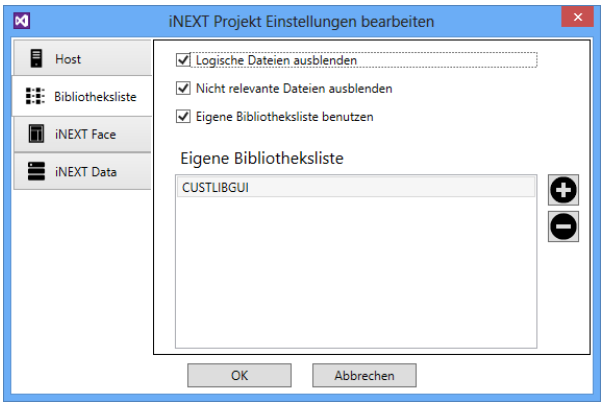



Im Einstellungen-Editor sind folgende Angaben erforderlich:

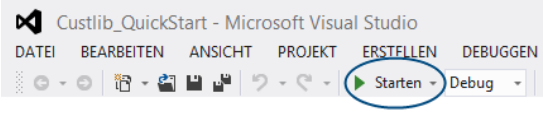
1. Host: IP-Adresse Ihrer AS/400, i5, System i o.ä.
2. User/Passwort können optional angegeben werden, wenn Sie eine programmgesteuerte Anmeldung wünschen



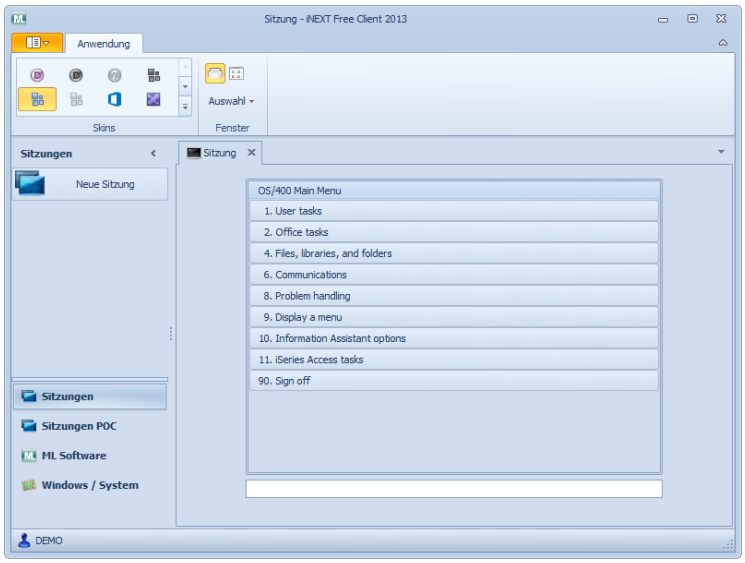

Darüber hinaus können eine Vielzahl weiterer Eigenschaft festgelegt werden. Empfehlenswert ist die Angabe einer Bibliotheksliste, wodurch der Ladeprozess bei Dateizugriffen beschleunigt wird.



Wenn die Einstellungen an Ihre Umgebung angepasst sind, dann klicken Sie in der Funktionsleiste vom Visual Studio zunächst auf „PROJEKT ERSTELLEN“ und dann auf „Starten“ und ihr Projekt wird kompiliert und gestartet.

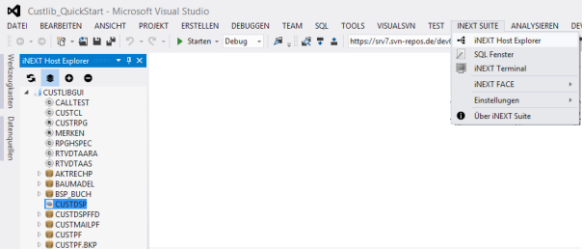



Im iNEXT 2013 Client können Sie nun mit einem Klick auf „Neue Sitzung“ eine Session auf Ihrer AS/400 öffnen und diese über die grafische Oberfläche vollumfänglich bedienen. Alle Funktionen stehen Ihnen zur Verfügung, ohne dass Sie dafür Änderungen auf der AS/400 Seite vornehmen müssen.

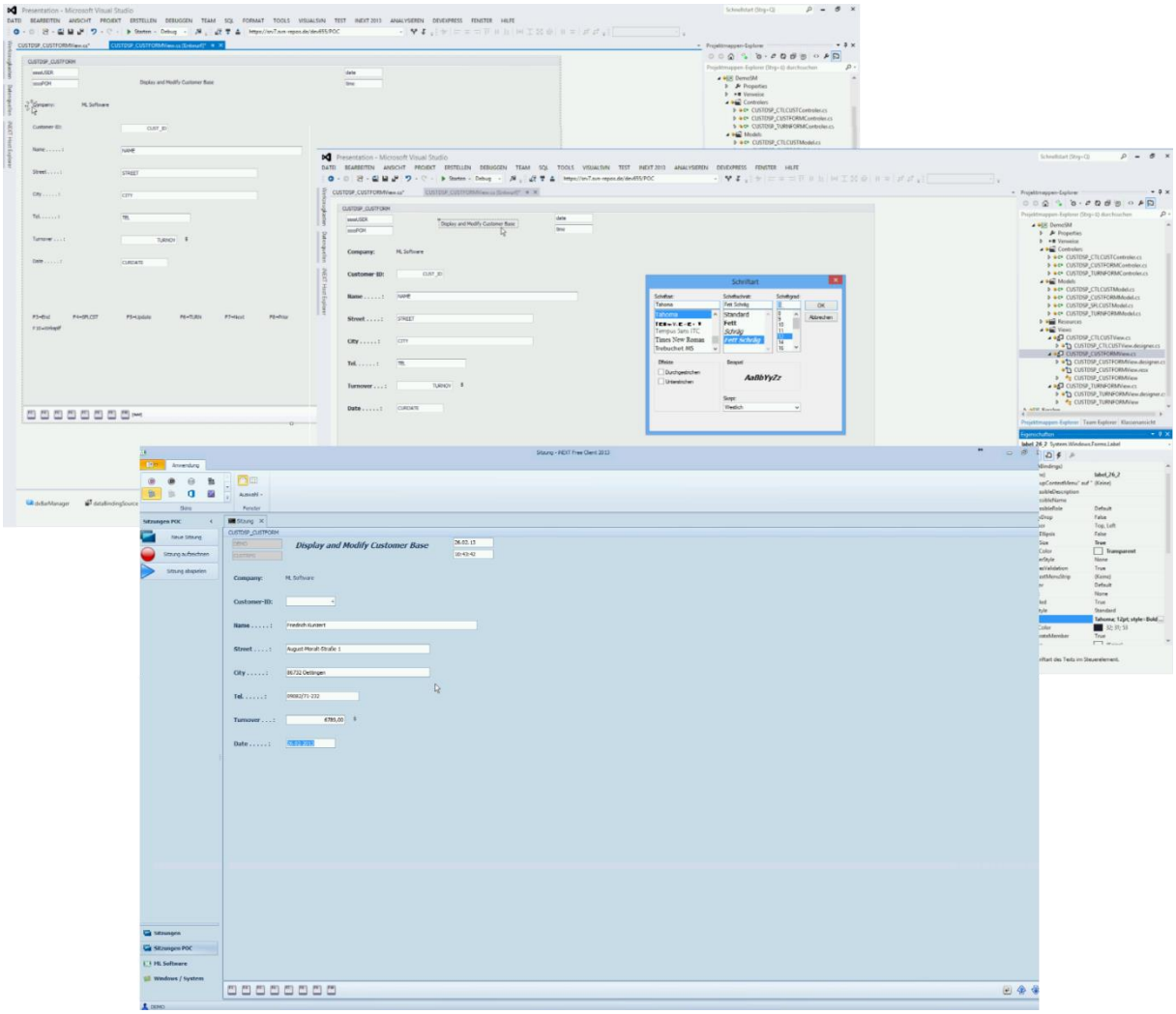


## Schritt 2: Migration der Displayfiles

Um Masken individuell zu gestalten, können die Displayfiles in .NET-Objekte migriert werden. Mit dem „iNEXT Host Explorer“ werden die vorhandenen RPG-Objekte angezeigt. Das gewünschte Displayfile wird ausgewählt und die Migration per Knopfdruck gestartet. Das Ergebnis sind Windowsoberflächen mit .NET-Objekten und Sourcen, die dann im Visual Studio zur individuellen Nutzung und Weiterentwicklung zur Verfügung stehen.

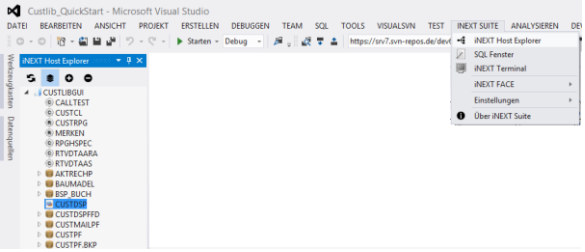



Die migrierten Objekte können z.B. per Drag & Drop neu angeordnet oder Fonts und Farben im Dialog verändert werden. Aus Subfiles erzeugt die Migration die windowstypische, tabellarische Darstellung, die z.B. schon die Veränderung der Spaltenanordnung und die auf- und absteigende Sortierung der Daten ermöglicht. Besonders nützlich sind die komfortablen Exportfunktionen, die das iNEXT 2013 für unterschiedliche Formate (Excel, PDF, RTF, CSV, HTML, Text) anbietet. Schnell und unkompliziert können so bereits funktionale Verbesserungen umgesetzt werden ohne dass die Performance des iNEXT 2013 Client darunter leidet.



## Schritt 3: Funktionale Erweiterungen

Eine rein optische Modernisierung der Anwendungen ist natürlich nur ein erster Schritt, um bestehenden IBM i Anwendungen eine sichere Zukunft zu eröffnen. Das Ersetzen der alten Green Screen-Oberflächen durch moderne, grafische Clients ist für die Anwender aber immer auch die schon auf den ersten Blick sichtbarste Verbesserung.



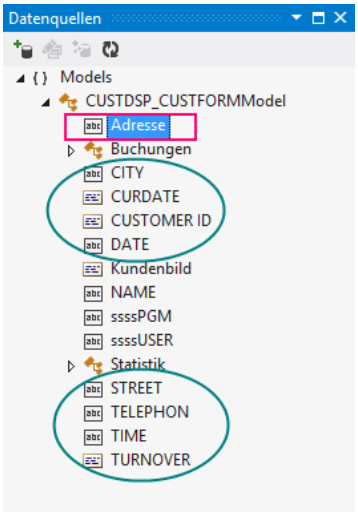

Den Möglichkeiten zusätzlichen Nutzen und Mehrwert mit funktionalen Weiterentwicklungen zu generieren sind im Visual Studio mit den leistungsfähigen iNEXT und .NET Technologien praktisch keine Grenzen gesetzt. Im Folgenden werden für einige typische Beispiele die Vorgehensweisen gezeigt, um sowohl die technische Machbarkeit als auch die Einfachheit zu demonstrieren.

### Definition von Feldern

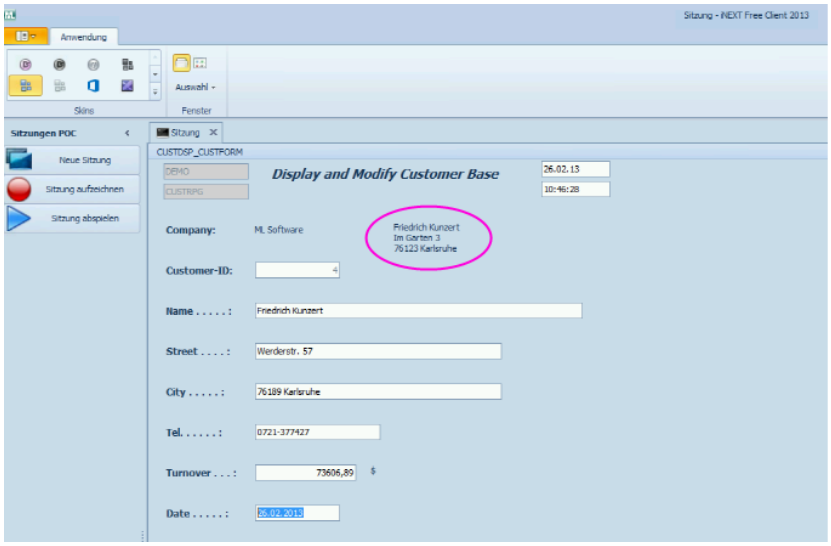
Das Beispiel zeigt die Definition eines neuen Feldes vom Typ „string“ mit dem Namen „Adresse“. Dieses Feld setzt sich aus mehreren Datenfeldern des RPG-Programmes zusammen, die durch einen Zeilenumbruch „\r\n“ voneinander getrennt sind.

```
public string Adresse
{
    get { return NAME + "\r\n" + STREET +  "\r\n" + CITY; }
}
```

iNEXT Suite 2013 arbeitet hierbei mit den identischen RPG-Feldnamen, nämlich „NAME“, „STREET“ und „CITY“. Diese und alle weiteren RPG-Felder (siehe grüne Umrandung im Screenshot) werden in den Datenquellen von Visual Studio zur Verfügung gestellt. Von dort können sie ausgewählt, per drag & drop auf einer Maske platziert und bereits mit einem passenden Darstellungstyp (z.B. Kalender für Datumsfeld) versehen werden.




Sobald das neue .NET-Feld deklariert ist, steht es ebenfalls in den Datenquellen vom Visual Studio zur Verfügung und kann in gleicher Weise wie die RPG-Felder verwendet werden.



## Einbindung von Bildern

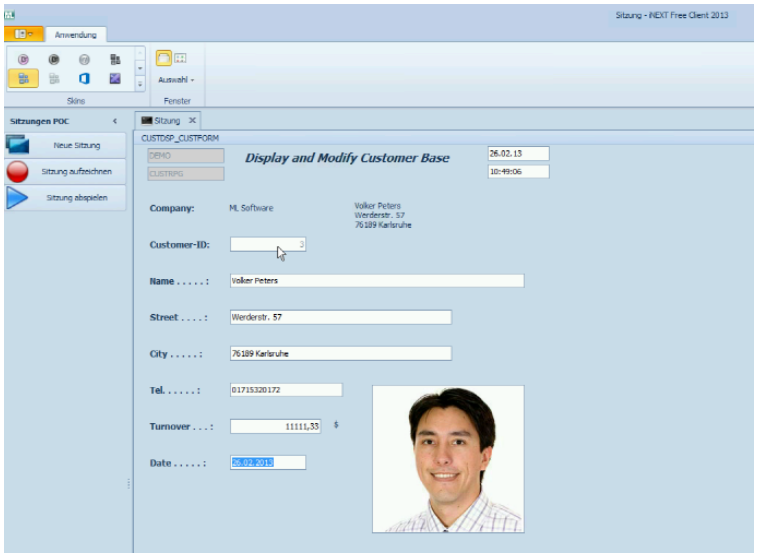
In der Praxis stehen Bilder häufig als Dateien (z.B. *.jpg) außerhalb vom System i zur Verfügung und können daher in Green Screen-Applikationen kaum oder nur mit großem Aufwand eingebunden werden. Im iNEXT 2013 Client ist eine Integration von Bildern vergleichsweise schnell und einfach umzusetzen.

Es wird ein neues Feld „Kundenbild“ vom Typ „Image“ erzeugt, das versucht, eine bestimmte jpg-Datei zurückzugeben. Mit dem Befehl „FromFile“ wird der Dateiname inklusive Verzeichnisangabe übergeben. Wird die Datei nicht gefunden, wird ein „null“ zurückgegeben, d.h. das Kundenbild bleibt leer.

```
public Image Kundenbild
{
    get
    {
        try
        {
            return Image.FromFile(@"C:\Source\POC\Presentation_ALL\Pictures\" + CUST_ID + ".jpg");
        }
        catch
        {
            return null;
        }
    }
}
```

Hinweis: In diesem Beispiel setzt sich der Dateiname (z.B. 1.jpg) aus der Kundennummer „Cust_ID“ und der Dateiendung „jpg“ zusammen, so dass hierüber die eindeutige Beziehung zwischen einem Kunden und seinem Bild hergestellt werden kann. Natürlich kann der Name der Bilddatei z.B. auch in der Kundendatenbank gespeichert sein.


Sobald das neue Feld deklariert ist, steht es wieder in den Datenquellen vom Visual Studio zur Auswahl und kann auf der Maske platziert und gestaltet werden.



## Integration zusätzlicher Daten

Das folgende Beispiel zeigt die Einbindung von Statistikdaten. Um diese zusätzliche Datentabelle von der DB2 oder auch aus anderen Datenbanken in die Maske einzubinden, wird per Mausklick ein neues Zugriffsobjekte erzeugt. Es erhält einen individueller Klassennamen z.B. „Statistik“, über den es dann im .NET-Quellcode angesprochen werden kann.

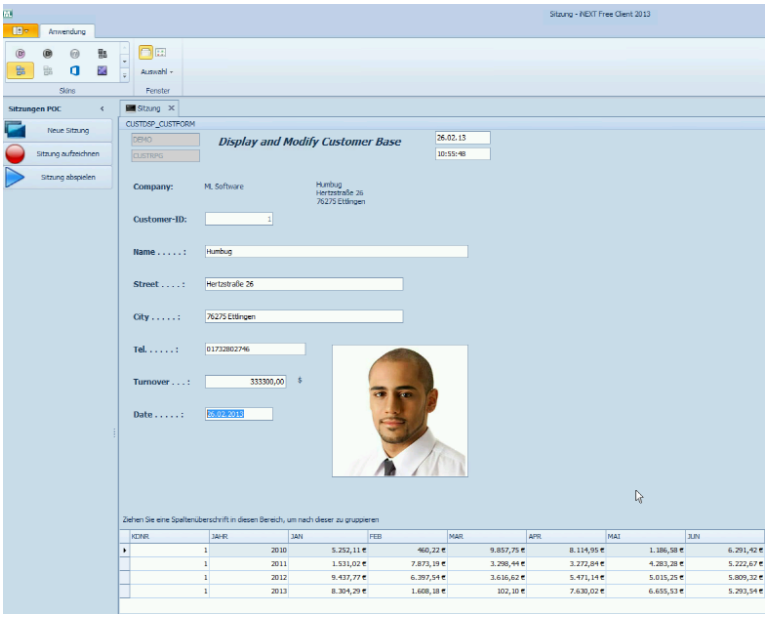
Im Quelltext erfolgt dann die Deklaration des neuen Feldes „Statistik“, das nun eine Datenmenge und nicht mehr nur einen einzelnen Wert enthält. Die Verbindung zwischen Kunden- und Statistikdaten wird über die die Felder „CUST_ID“ und „CurStatId“ hergestellt.

```
public List<statistik> Statistik
{
    get
    {
        if (CUST_ID != CurStatId)
        {
            CurStatId = CUST_ID;
            CurStatistik = statistikAccess.GetSingleTable(conn, 0, " where KDNR = " + CUST_ID);
        }
        return CurStatistik;
    }
}
```

Die Verknüpfung zur Datenbank erfolgt anschließend über die DataBindingSource. Damit wird sichergestellt, dass für jeden Kunden auch nur seine zugehörigen Statistikdaten angezeigt werden.

```
private void dataBindingSource_CurrentChanged(object sender, System.EventArgs e)
{

    statistikBindingSource.DataSource = Controler.Data.Statistik;

}
```

Das Ablegen und Platzieren der Statistik-Daten erfolgt über die Datenquellen in Form eines Datagrid. Die Eigenschaften des Datagrid können beliebig angepasst werden. In der Anzeige können Zahlen als Geldbeträge mit Währungsangabe dargestellt und mit einer Taschenrechnerfunktion versehen werden, Datumsfelder in einem bestimmten Format angezeigt und über einen Kalender ausgewählt werden u.v.m.



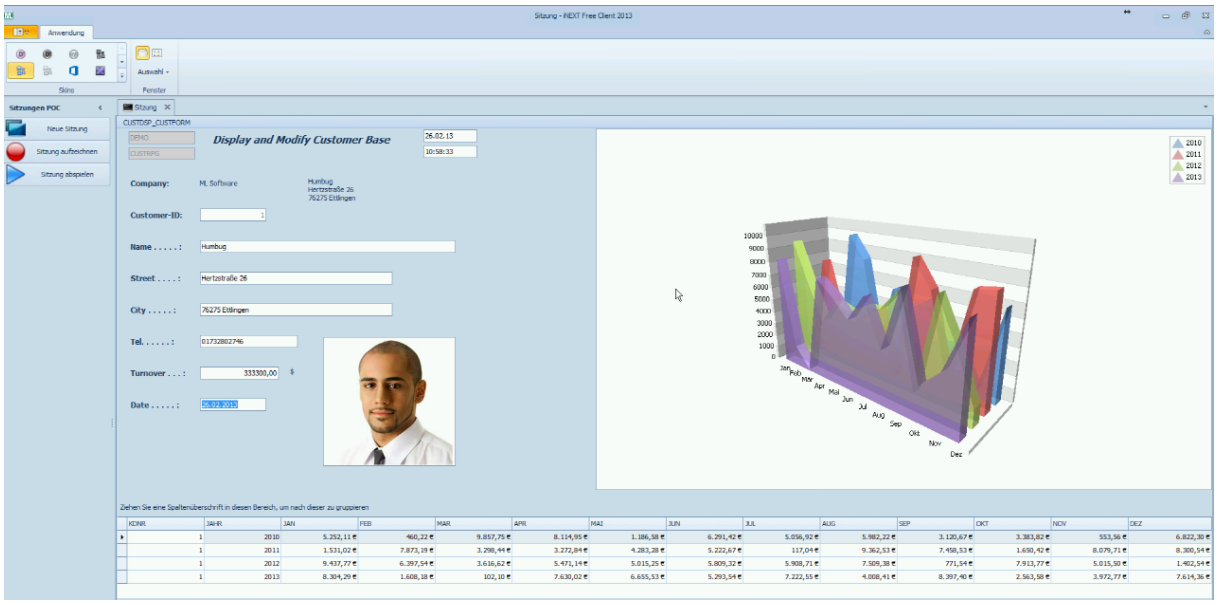
## Visualisierung von Daten

Zur grafischen Darstellung von Daten steht unter anderem die „Chart“-Komponente zur Verfügung. Sie wird auf der Maske platziert und muss dann nur noch mit den gewünschten Daten gefüllt werden.

Hierzu wird das bereits für die Datensynchronisation verwendete Ereignis „Current_Change“ erweitert. Das Codebeispiel zeigt, wie für jeden Monat ein neuer Datenpunkt in der Grafik erzeugt, beschriftet und mit dem entsprechenden Wert aus der Statistik gefüllt wird.

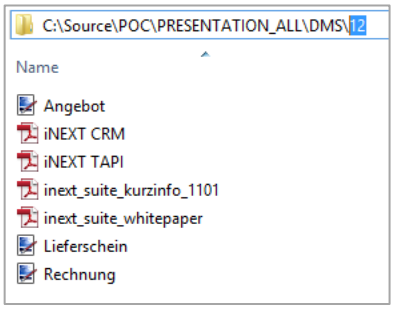
```
private void dataBindingSource_CurrentChanged(object sender, System.EventArgs e)
{
    chartControl1.Series.Clear();
    if (Controler.Data.Statistik != null)
    {
        foreach (statistik row in Controler.Data.Statistik)
        {
            Series series = new Series(row.JAHR.ToString(), ViewType.Area3D);

            series.Points.Add(new SeriesPoint("Jan", row.JAN));
            series.Points.Add(new SeriesPoint("Feb", row.FEB));
            series.Points.Add(new SeriesPoint("Mär", row.MAR));
            series.Points.Add(new SeriesPoint("Apr", row.APR));
            series.Points.Add(new SeriesPoint("Mai", row.MAI));
            series.Points.Add(new SeriesPoint("Jun", row.JUN));
            series.Points.Add(new SeriesPoint("Jul", row.JUL));
            series.Points.Add(new SeriesPoint("Aug", row.AUG));
            series.Points.Add(new SeriesPoint("Sep", row.SEP));
            series.Points.Add(new SeriesPoint("Okt", row.OKT));
            series.Points.Add(new SeriesPoint("Nov", row.NOV));
            series.Points.Add(new SeriesPoint("Dez", row.DEZ));
            chartControl1.Series.Add(series);

        }
    }
}
```

Die Grafik kann z.B. auch gedreht, vergrößert und verkleinert, gedruckt und exportiert werden, je nachdem wie der Anwender die Anzeige wünscht, um die Daten optimal darzustellen.



## Einbindung von Dokumenten

Viele Dokumente werden in den Unternehmen heute als Dateien auf unterschiedlichen Servern gesichert. Diese pdf-, jpg- oder sonstigen Dateien können ähnlich wie die Bilder mit einem iNEXT 2013 Client in eine IBM i Anwendung eingebunden werden. Das folgende Beispiel nutzt eine eindeutig definierte Verzeichnisstruktur. Für jeden Kunden gibt es einen Ordner, dessen Bezeichnung der jeweiligen Kundennummer entspricht. Hier sind dann die Dokumente für diesen Kunden abgelegt.




Auf der Maske wird das DocViewer-Control abgelegt. Zur Synchronisation der Daten wird wieder das Ereignis „CurrentChanged“ genutzt. Dem DocViewer wird hier das Dokumentenverzeichnis, das sich aus einem festen Pfad und der individuellen Kundennummer „CUST_ID“ zusammensetzt, mitgeteilt.

```
private void dataBindingSource_CurrentChanged(object sender, System.EventArgs e)
{
    docView1.Uri = @"C:\Source\POC\Presentation_ALL\DMS\" + Controler.Data.CUST_ID;
}
```

Die gefundenen Dateien werden in einem Preview als Dokumentenkarussell oder auch in Listenform angezeigt. Der Anwender kann durch die Dokumente blättern, ein Dokument auswählen, groß anzeigen, separat positionieren, auf Wunsch per Knopfdruck exportieren u.v.m.

## Vorher-Nachher-Betrachtung

Der Vergleich zwischen der GreenScreen-Anwendung und dem erweiterten iNEXT 2013 Client zeigt, welche Verbesserungen mit den technologischen Möglichkeiten von .NET und iNEXT in kurzer Zeit erzielt werden können. Der Nutzen, den die Anwender bei ihrer täglichen Arbeit aus den neuen Funktionen, der komfortableren Bedienung und der ansprechenderen Optik ziehen, steigert den Mehrwert und die Zukunftsfähigkeit der modernisierten Lösung.

Anhand des Bildes (s.u.) erkennen sie die Entwicklung von der einfachen GreenScreen-Anwendung über das risikofreie GUI-on-the-fly bis hin zum erweiterten iNEXT 2013 Client. Um zu erleben, wie schnell und effizient diese Weiterentwicklungen in der Praxis sind, empfehlen wir Ihnen den Download unserer iNEXT 2013 Präsentation. In dem ca. 20 min. Video können Sie jeden der hier vorgestellten Schritte an Ihrem Bildschirm in Echtzeit mitverfolgen.

# ML-Software GmbH

Schon Mitte der 90-er Jahre hat sich die ML-Software auf die Modernisierung damaliger AS/400 Anwendungen spezialisiert. Von Beginn an wurden innovative Konzepte und Produkte verwirklicht, die den bewährten aber in die Jahre gekommenen AS/400-Anwendungen wieder Zukunftsfähigkeit und essentiellen Zusatznutzen brachten. Bekannte Unternehmen aus unterschiedlichsten Branchen wie Adelholzener, Bison, Hauck, Hörmann, KAPS, Meggle, Sanetta, Yamaha u.v.m. gehören zu unseren Kunden.

Heute bietet ML-Software sehr umfassendes und fundiertes Wissen darüber, wie Sie Ihre bewährten IBM i Lösungen in Verbindung mit den modernen .NET-Technologien bedarfsorientiert, sicher und vor allem effizient weiterentwickeln, durch neue .NET-Programmierung funktional erweitern und durch Integration externer Lösungen komplettieren. Natürlich können wir Ihnen dieses Wissen auch für die Reproduktion Ihrer IBM i Lösungen unter .NET zur Verfügung stellen. Machen Sie hiervon Gebrauch und bewerten Sie Ihre vorhandenen Hard- und Softwarelösungen neu, anhand dessen, was iNEXT Suite 2013 daraus machen. Sie erreichen Ihre Ziele auf diesem Weg nicht nur schneller, kostengünstiger und ohne Risiko, sondern müssen auch keine Abstriche bei der Qualität machen.

Als erster Hersteller haben wir bereits 2010 grafische .NET-Clients für IBM i und ihre Vorgänger ins Internet gestellt. Der iNEXT Free Client und der iNEXT Sample Client können absolut kostenfrei heruntergeladen und genutzt werden. Hiermit stellen wir unter Beweis, dass mit iNEXT Suite die grafische Oberfläche lediglich der Anfang für eine zukunftsorientierte IT-Strategie ist. Wir bieten Ihnen außerdem ein kostenfreies Proof-of-Concept an, mit dem wir Ihnen nachweisen, dass Ihre anstehenden IT-Aufgaben mit unseren Technologien und unserem Know How erfolgreich gemeistert werden.

ML-Software GmbH
Hertzstr. 26
76275 Ettlingen

Tel. +49 (0) 7243 – 56550
Fax +49 (0) 7243 – 565516

info@ml-software.com

www.ml-software.com

www.inextsuite.com