

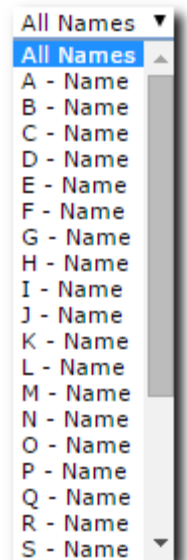
OnePoint Portal and Developer Toolkit

Portal

OnePoint Portal was conceived as a framework for Relational Data and our business partners to develop and deploy broadly-scoped web applications, where many applications would require shared infrastructure and services. The Portal is used to manage users, authentication, work areas, menus, application definitions, and provides a browser user interface for launching multiple stateful applications.

The need for a product such as OnePoint Portal arises when an organization intends to deploy numerous, broadly-scoped and possibly multiple types of applications for multiple types of users – think ERP class systems. Such systems may entail hundreds or even thousands of database tables across a variety of organizational interests. In such cases, applications may require certain infrastructure and services, which may be shared by each. OnePoint Portal provides that type of infrastructure and those types of services.

Sundry applications and products may be deployed under OnePoint Portal, only limited by one's imagination and business requirements. It's possible that organizations may define and deploy thousands of menu items for interactive applications and reports under OnePoint Portal for unlimited numbers of users and groups.



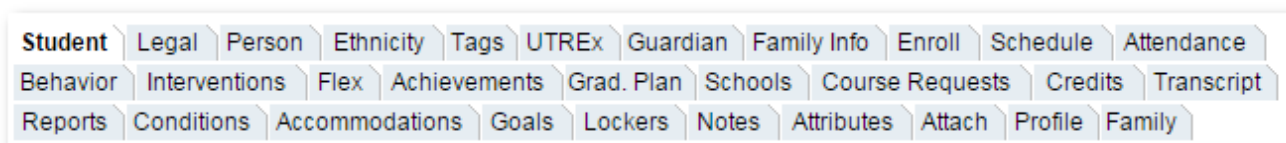
Developer Toolkit

We offer an array of application development tools as well as a database inquiry and maintenance model. The word "model" in this context means a design pattern and code basis for web applications that may support many functional areas. We have come up with one which is more functional and adaptable than anything else we have seen. It is only limited by the number of tables and views in your database and your imagination.

As we break down the model into individual components for analysis and documentation, please keep in mind that each component works together in a cohesive whole, and are viewed together as a single container (page), which will be shown further along.

Tabs

Tabs are a way of both separating and organizing related data into a cohesive container. In a robust student information system, many database tables exist which are related to and dependent upon a "student".




Content underneath the tab bar changes as users navigate to each tab. Tabs automatically adjust their position within the tab bar according to screen-width. The tab bar may grow vertically as screen-width



is decreased.

A "Tab Bar" is paired with an "Identity Bar" which names or describes the current row of a database table or SQL view (a student name in this case).



The Identity Bar includes "next" and "prior" buttons  to navigate from row to row. A school counselor may be viewing a student's class-schedule tab, and use "next" and "prior" buttons to navigate a student list, while displaying class-schedule content for the currently selected student.

Hierarchical Data Relationships

The class-schedule tab may include additional tabs underneath for drilling down into data such as class assignments, and utilities such as grade calculations. There is no defined limit to the drill-down hierarchy. Consider a hierarchy such as School > Year > Students > Class Schedules > Class > Assignments, where lower-level entities are dependent on those above.

Record Frame

The "record frame" is used for viewing, adding, and modifying individual records, which may be from one or more database tables: a student, and a school enrollment record, for example.

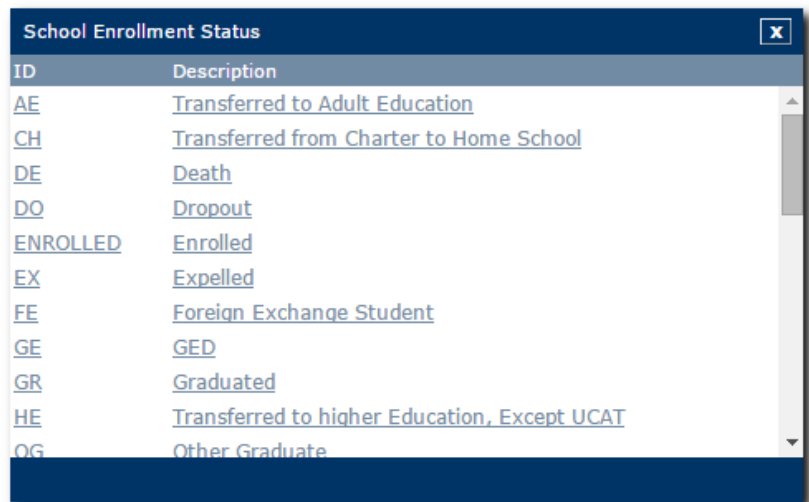
A screenshot of a web-based form titled "Student Record" and "Enrollment Record". The "Student Record" section contains fields for Student ID, Person ID, First Name, Middle Name, Last Name, Name Suffix, Preferred Name, Birth Date (with a calendar icon and "(YYYY-MM-DD)" text), Birth Place, Gender (with "M/F" text), Current Grade (with a red arrow and a calendar icon), State/Other ID, and Record Status (with a calendar icon). The "Enrollment Record" section contains fields for Enrollment Status (with a red arrow and a calendar icon) and Grade (with a red arrow and a calendar icon). The form has a light blue background and a vertical scrollbar on the right.

Focus remains on the form as the tab key is pressed to navigate from one input element to the next. Required fields are indicated by the red arrow. Try submitting a record with missing fields and required input element will light up, with an error message displayed.

Popup Lists

Input elements may be filled in by selecting from popup lists, indicated by an icon adjacent to the field.


Clicking a list item returns a code value to an input element.

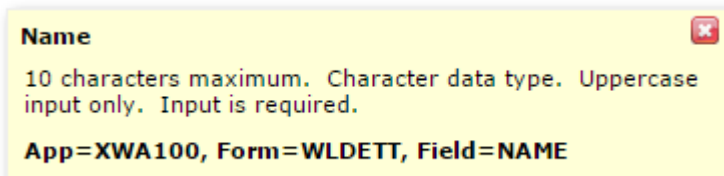


ID	Description
AE	Transferred to Adult Education
CH	Transferred from Charter to Home School
DE	Death
DO	Dropout
ENROLLED	Enrolled
EX	Expelled
FE	Foreign Exchange Student
GE	GED
GR	Graduated
HE	Transferred to higher Education, Except UCAT
OG	Other Graduate

Help and Tips

Form help and field tips can be

activated by clicking the  icon.



Name

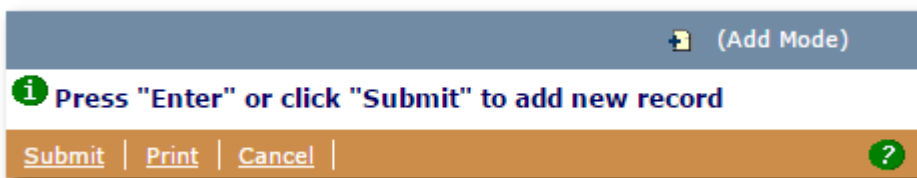
10 characters maximum. Character data type. Uppercase input only. Input is required.

App=XWA100, Form=WLETT, Field=NAME

Default "help" content is generated by the application generator as a placeholder. A program is provided to maintain it. Help text changes as users tab from field to field.

Record Action and Message Bars

A record frame is paired with record action and message bars. Available record actions such as "Submit" or "Cancel" mutate, depending on record mode.



(Add Mode)

i Press "Enter" or click "Submit" to add new record

Submit | Print | Cancel | ?

Record modes include View Mode, Add Mode, Copy Mode, Change Mode, and Delete Mode. The currently selected mode, including a mutable icon associated with it, is shown immediately above the message bar (Add mode in this case). Default actions include Print, Add, Copy, Change, and Delete. Additional actions, like Auto Fill, may be added to the action bar as required.

The model includes a set of default prompts, error and status messages, which may be extended and tailored by developers, which are shown in the message bar. Message types include Information, Caution, and Error. Each type has a corresponding icon image associated with it.

Several keyboard short-cuts facilitate Add, Copy, Change, and Delete operations.

- F2: Activate Change Mode
- F3: Activate Copy Mode
- F6: Activate Add Mode
- Delete: Activate Delete Mode
- Esc: Cancel Add, Change, Delete and Return to View Mode
- Enter: Submit Additions, Updates, and Deletions
- Ctrl-P: Print Record

List Frame

Browsing a list of records helps put data in context for inquiry purposes, and helps select records for maintenance purposes.

The list may include multiple columns (often shown below the record frame), or as in this example, may consist of a single column placed on the left or right of the record frame. Changing the order of a list is generally done by clicking a column heading. Sort orders may include multiple columns, such as students within grade levels. A status bar shows the currently selected row, along with the total number of rows downloaded. A link labeled More appears if more rows are available for download. Rows are typically retrieved in increments of 100 and appended to the bottom of the list.

Keyboard and Mouse Short-Cuts For List Frames

Several keyboard short-cuts assist with navigating the list frame and selecting rows for change and delete operations.

- Home: Go to and Select First Row
- End: Go to and Select Last Row
- PgDn: Scroll List Forwards



Student Name / Grade Level
Aaron, Aidan (9)
Aaron, Kyohei (8)
Abbott, Adam (9)
Aboagye, Gareth (8)
Adair, Ivy (8)
Alexander, Adhis (8)
Allredge, Brenner (8)
Alley, Bronsen (8)
Almonroeder, Amado (9)
Almonroeder, Bryant (12)
Alvarez, Carston (12)
Anderl, Allayna (8)
Anderon, Chet (8)

- PgUp: Scroll List Backwards
- Arrow-Down: Go to and Select Next Row
- Arrow-Up: Go to and Select Prior Row
- Shift-Arrow-Down: Multi-Select Next Row
- Shift-Arrow-Up: Multi-Select Prior Row
- Ctrl-A: Select All Rows
- Ctrl-Shift-Arrow-Up: Select All Rows From Current to First
- Ctrl-Shift-Arrow-Down: Select All Rows From Current To Last
- Shift-Click: Select Range of Rows From Current to Clicked
- Ctrl-Click: Add Clicked Row to Multi-Selection List

The currently selected record is retrieved and viewed in the record frame. Users can activate Change or Delete modes after selecting one or multiple rows.

Named Filters

One option for limiting list content without programming is through the configuration of predefined filters, which appear as combo box entries.


Selecting one of the named filters from a combo box causes the list to be filtered and refreshed. Selecting "A - Name" in this example displays only students having a last name beginning with the letter "A". Various SQL expressions may be included in named filters.

Filter Dialog

The model includes a dialog that enables users to filter lists, by dynamically generating expressions which may include moderately complex SQL selection criteria.


The screenshot shows a 'Filter' dialog box with a title bar and a close button. Below the title bar are 'Submit' and 'Clear' buttons. The main area contains several rows of input fields and dropdown menus:

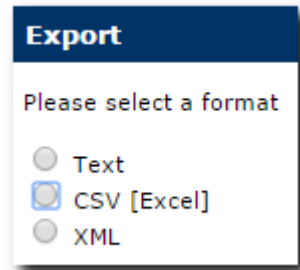
- Student ID: [text input] Is Equal To [dropdown]
- Person ID: [text input] Is Equal To [dropdown]
- First Name: [text input] Is Equal To [dropdown]
- Middle Name: [text input] Is Equal To [dropdown]
- Last Name: [text input] Is Equal To [dropdown]
- Name Suffix: [text input] Is Equal To [dropdown]
- Preferred Name: [text input] Is Equal To [dropdown]
- Birth Date (YYYY-MM-DD): [text input] and [text input] Is between [dropdown]
- Birth Place: [text input] Is Equal To [dropdown]

The filter dialog may be activated by clicking the filter icon  or pressing the F7 key.

Export Dialog

The model includes a dialog that enables users to export and download list data, using various formats, including plain text, comma delimited text, and XML.

An export dialog may be activated by clicking the export icon  or pressing the F8 key.



Overview

The following screen-shot shows a model-based application within the context of a school district, a school, a school year, and a grading term. Changing one of the combo box items causes the student list and all related data underneath it to be refreshed to reflect the selected context.

The screenshot displays a web-based application interface for school navigation. At the top, the title is "School Navigation [SSCH.201]". Below the title, there is a breadcrumb trail: "Work Areas > Student Info > SIS Admin". On the right side of the top bar, there are "Exit" and "Logout" links. The main navigation area includes dropdown menus for "District" (Weilenmann), "School" (Jr. High (7-9)), "Year" (2013-2014), and "Term" (QT2), along with a "Set as default" link. Below this, there are buttons for "Home", "Students", and "Student School Info".

The main content area is titled "Abbott, Adam [0000000310]". It features a horizontal menu with various tabs: Student, Legal, Person, Ethnicity, Tags, UTREx, Guardian, Family Info, Enroll, Schedule, Attendance, Behavior, Interventions, Flex, Achievements, Grad. Plan, Schools, Course Requests, Credits, Transcript, Reports, Conditions, Accommodations, Goals, Lockers, Notes, Attributes, Attach, Profile, and Family. Below the menu, there is a "All Names" dropdown and a "(Change Mode)" link.

The interface is divided into two main sections. On the left, there is a list of students with "Abbott, Adam (9)" selected. On the right, there is a "Student Record" form. The form includes fields for Student ID (0000000310), Person ID (0000001323), First Name (Adam), Middle Name (Egnew), Last Name (Abbott), Name Suffix, Preferred Name (Abbott, Adam), Birth Date (1999-08-07), Birth Place, Gender (F), Current Grade (9), State/Other ID (1111972), and Record Status (INACTIVE). Below the Student Record, there is an "Enrollment Record" section with fields for Enrollment Status (ENROLLED) and Grade (9).

This is an example of cohesive and productive database inquiry and maintenance.

Flexible User Interface

The next screen-shot shows the same model, but with a different look and a different application, which speaks to its flexibility.



A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Program: Type: Patron: Cash Account:

Abbott, Adam Egnew Has not requested participation

Jr. High (7-9) Abbott Family | .00 ▾

Grade: 9

Jr. High Art	Performing Arts Intermediate	Middle School Athletics
Band	Basketball summer camp	Biology
LPA Calendar	Cap and Gown	Insufficient Funds
Choir	Citizenship Make-up	Team Clothing
SBO/Chess/Jr. Honor	Jr. High CTE	Elite
Directory	Ballroom	Dance company dress fee
Earth systems science	Performing Arts Advanced	Full year debate team
Cell Phone	Donation	Half year debate team
Library Fee	School lunch	Instrument Rental
Elementary Musical	Orchestra	Music Uniforms
Fit For Life	Registration	Sweaters
Schedule Change	Social Dance	Spiritwear
Team Sports	Missing Damaged Textbooks	3D Art
7th Grade integrated science	8th Grade integrated science	9th Grade Lab Fee

Cash Tended: **\$0.00**

(View Mode)

TC	Account	Amount
SA		25.00

Row 2 of 3

Seq	TC	Account	Item ID	RC	Amount
1	SA		ART	FP	15.00
Jr. High Art					
2	SA		ARTS	FP	25.00
Performing Arts Intermediate					
3	SA		BAND	FP	10.00
Band					

Change Due: \$-50.00

This shows the model deployed in a point-of-sale application, proving its adaptability from inquiry and maintenance to transaction entry and processing, yet sharing much of the same code base.

Model Summary

This database inquiry and maintenance model, combined with utilities such as popup list and date selections, provide a well defined design and code basis, which can be extended to any context or business function. Models and utilities provide a significant productivity boost to developers. End-users benefit by having a highly-functional user interface which is consistent across many applications.

Application Generator

This model is a basis for our application generator, which is included with your license to OnePoint



Portal and Developer Toolkit. The application generator will, with a single command and easily modified script, generate the HTML, Javascript, and application code, along with definitions for the tabs, popups, help text, and menu item placement. Applications generated using the application generator are immediately fully functional as described above, or can easily be modified for a more customized experience.

Source code for generated applications is divided into simple to understand modules that separately deal with browser interactions and database interactions. Again, the source code is easy to read and easy to customize.

We anticipate that the application generator will be very valuable, as it makes it possible to quickly respond to the needs of your patrons, and will reduce the time and money spent on development.

Event Handlers and Database Driven Validation

The OnePoint Portal and Developer Toolkit includes tools that allow you to place referential-integrity constraints, business rules, and other logic incidental to database operations into defined database event handlers. Database event handlers export a set of procedures that can be configured to run before or after read, write, update, and delete operations occur on a file, regardless of the application that is making the modification.

Database event handlers can be written independently of user-interface and other application components. If written in conjunction with other components, then database event handlers might be written before all others, and are the best way to ensure the validity and integrity of data, to prevent orphaned records in databases that define parent-child relationships, and to implement a "single source of truth" (so to speak), in regard to data-validation and other types of rules. They can be used to effectively prevent unwanted updates from unauthorized interfaces.

Our database event handlers include a framework that significantly streamlines the programming interface, makes them easy to deploy, and provides functionality that is helpful. They are a valuable part of an overall Web application framework that can be used to develop robust, broadly-scoped applications.

Application code created using our application generator will interface automatically with defined database event handlers, and a command is included that automatically generates the source code for database event handlers based on the file of interest.

Report Utility and PDF Generator

The OnePoint Portal and Developer Toolkit includes a report utility and PDF generator to help developers quickly build graphical PDF reports using techniques similar to those that they will use developing web applications.

Reports are deployed as menu items or in tabs in the user interface. They can be created quickly, and can be customized to accept input from the end user.

Email Server

An e-mail server is included with your license, along with APIs for sending emails from the applications that you design. OnePoint Portal interfaces with the email server for sending password reset emails to users that have lost their password, and several student system applications interface with the e-mail server for sending messages to parents, students, teachers, or administrators.