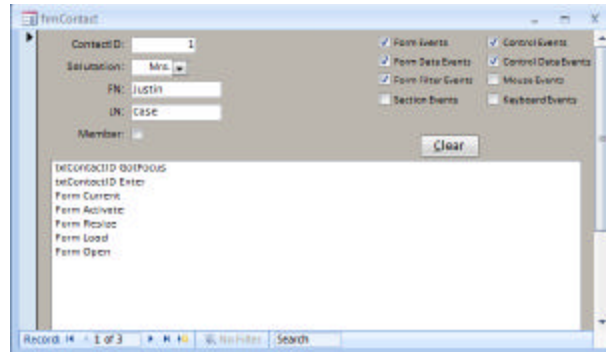


# Event Demonstrator



## Form Events

- **Current** (Callahan Chapter 2)
  - occurs when user moves to another record and when the form first opens and shows its initial record
  - use to execute code whenever a record is displayed
- **AfterUpdate** (Callahan Chapter 2)
  - occurs after a record has been saved
- **Dirty** (we used in Callahan Chapter 2 WebCast)
  - occurs when user changes data displayed on a bound form
- **Undo** (we used in Callahan Chapter 2 WebCast)
  - occurs when user returns a form's data to its original state
- **KeyDown** (Callahan Chapter 4)
  - occurs when user presses a key while a form has the focus, before the key is sent to the control

## Form Events

- **BeforeUpdate** (Callahan Chapter 4)
  - occurs before a new or modified record is saved
  - for custom validation: VBA code can check the data entered, then either allows record to be saved or cancels save & requires corrective action
- **Filter** (Callahan Chapter 3)
  - occurs when the user selects Filter By Form or Advanced Filter/Sort
- **ApplyFilter** (Callahan Chapter 3)
  - occurs when the user selects Toggle Filter or Filter By Selection or when setting form's Filter property or FilterOn property via macro/VBA
    - i.e., whenever the set of records is about to change
- **Error** (Callahan Chapter 7)
  - occurs when an error occurs while a form has the focus
  - triggered by Jet database engine errors but not by VBA runtime errors

## Form Events

- **BeforeInsert** (Callahan Chapter 4)
  - occurs just before user types the first character in a new record, but before the record is actually created
  - we'll use to fix a problem with AllowEdits while adding a record via our read-only Contacts form
- **AfterInsert**
  - occurs after a new record has been saved to disk
- **Open** (Callahan Chapter 9)
  - occurs when a form is opened, but before the first record is displayed
- **Activate**
  - occurs when the form receives the focus & becomes the active window
- **Deactivate**
  - occurs when the form loses the focus
- **Close** (Callahan Chapter 9)
  - occurs when when a form is closed and removed from the screen

## Control Events

- Available events vary by type of control
- Click (Callahan Chapter 2, 3, 4, 6, ...)
  - not just for command buttons!
- Change
  - occurs when the contents of a text box or the textbox portion of a combo box changes
- AfterUpdate (Callahan Chapter 3, 4, 6)
  - occurs after a control's data has been updated, and just before the control is exited
- GotFocus
  - occurs when a control receives the focus
- LostFocus
  - occurs when a control loses the focus

## Control Events

- Enter
  - occurs before control receives focus from another control **on same form**
    - does not occur when a form regains the focus from another form
  - occurs before GotFocus
  - we'll use in Callahan 3 to change Contact form's Allow Edits property just before unbound combo box or filter option group receives the focus
- Exit
  - occurs before a control loses the focus to another control **on same form**
    - does not occur when a form loses the focus to another form
  - occurs before LostFocus
  - we'll use in Callahan 3 to change Contact form's Allow Edits property just before unbound combo box or filter option group receives the focus
- KeyDown (Callahan Chapter 4)
  - occurs when user **presses** a key while a control (or form) has the focus

## Sequence of Events

- Event Demonstrator.accdb
  - frmContacts
  - check boxes to control which events are logged
- Opening a form
  - Open(form)→Load(form)→Resize(form)→Activate(form)→Current(form)  
→Enter(1<sup>st</sup> control)→GotFocus(1<sup>st</sup> control)
- User presses Tab to go to next control on **same form**
  - Exit(1st control)→ LostFocus(1st control)→  
Enter→(2nd control)→GotFocus (2nd control)
- User types 3 characters into text box, then presses Tab to go to next control on same form
  - Dirty(form) → Dirty(control) → Change(for each character) →  
BeforeUpdate(control) → AfterUpdate(control) → Exit(1st control)→  
LostFocus(1st control) → Enter→(2nd control)→GotFocus (2nd control)

## Sequence of Events

- User saves with Records | Save
  - BeforeUpdate(form)→ AfterUpdate(form)
- User navigates to next record
  - Exit(control) → LostFocus(control) → Current(form) → Enter(control) →  
GotFocus (control)
- Shifting focus to a control on another open form
  - LostFocus(control)→Deactivate(form 1)→Activate(form 2)→  
GotFocus(control)
- User clicks in a combo box, selects a different row, then uses Undo to abandon the change
  - Exit(control) → LostFocus(control) → Enter(combo) →  
GotFocus (combo) → Dirty(form) → Dirty(combo) →  
BeforeUpdate(combo) → AfterUpdate(combo) → Click(combo) →  
Change(combo)→ Undo(form)

## Sequence of Events

---

- User closes a form
  - Exit(control) → LostFocus(control) → Unload(form) → Deactivate(form) → Close(form)
- Section Events
  - a few are available, but they're rarely used
- Mouse Events
  - MouseDown, MouseMove, MouseUp, MouseWheel
  - mouse coordinates (X,Y)
  - which mouse buttons are pressed
  - whether [Shift], [Ctrl], [Alt] keys are pressed
- Keyboard Events
  - Callahan Chapter 4
  - on your own...

## Engineering

---

- How does the list box show each event's name and other details to log all those different events?
  
- How does the Clear button wipe out the list box's contents?

## Practice Time

<input checked="" type="checkbox"/> Form Events	<input type="checkbox"/> Control Events
<input checked="" type="checkbox"/> Form Data Events	<input type="checkbox"/> Control Data Events
<input type="checkbox"/> Form Filter Events	<input type="checkbox"/> Mouse Events
<input type="checkbox"/> Section Events	<input type="checkbox"/> Keyboard Events

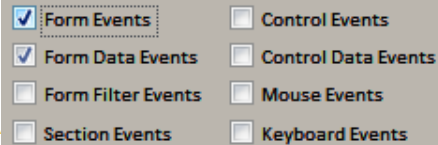
- Explore **form-level** events that fire when you create and save a **new record**.
  1. Adjust the event logging option check boxes to match those illustrated.
  2. Clear the event history list box so you're not distracted by older events.
  3. Use `frmContact` to create a new record, then pick a salutation, and type values for LN and FN fields.
  4. Save your new record.
  5. List the sequence of form-level events that occurred.
    - pay particular attention to event names that contain **Ins**

## Practice Time

<input checked="" type="checkbox"/> Form Events	<input type="checkbox"/> Control Events
<input checked="" type="checkbox"/> Form Data Events	<input type="checkbox"/> Control Data Events
<input type="checkbox"/> Form Filter Events	<input type="checkbox"/> Mouse Events
<input type="checkbox"/> Section Events	<input type="checkbox"/> Keyboard Events

- Explore **form-level** events that fire when **data errors** occur. [Hint: 3316 and 2237 are in your future.]
  1. Retain the event logging option check boxes illustrated.
  2. Clear the event history list box so you're not distracted by older events.
  3. Delete the FN field's value. (FN has **Is Not Null** as its **Validation Rule**)
  4. Attempt to save the record. Click OK in response to the message.
  5. List the sequence of form-level events that occurred.
  
  6. Press [Esc] to restore the record's data.
  7. Clear the event history list box so you're not distracted by older events.
  8. Click the Salutation combo box, then type `emperor`, then press [Tab]. Click OK in response to the message.
  9. List the sequence of form-level events that occurred. What number occurred this time?
  10. Press [Esc] to restore the record's data.

## Practice Time



- Three form-level events relate to deleting a record. Discover them by performing these steps.
  1. Retain the event logging option check boxes as illustrated above.
  2. Clear the event history list box so you're not distracted by older events.
  3. Use `frmContact` to attempt to delete any record.
  4. When the warning message appears, click No (don't delete the record).
  5. What sequence of form-level events occurred?
    - pay particular attention to event names that contain **Del**
    - what **Status** value was returned?
  6. Attempt to delete the record again.
  7. When the warning message appears, click Yes (do delete the record).
  8. What sequence of form-level events occurred?
    - pay particular attention to event names that contain **Del**
    - what **Status** value was returned this time?

## Practice Time



- Three form-level events relate to deleting a record
  - Delete(form) → BeforeDelConfirm (form) → AfterDelConfirm(form)
- Look up each event in online Help & learn more.
  1. Open online Help.
  2. Set Search scope to **Developer Reference** (illustrated in yellow below).
  3. Read each carefully up to the **Remarks** section (then quit or skim rest)

