

# Validating Data in Input Pro

a solution for ActiveX/COM developers

The logo for FarPoint, featuring the word "FarPoint" in a serif font. The letter "o" in "Point" is replaced by a blue, 3D-rendered sphere with a white highlight and a soft blue glow around it.

FarPoint

This white paper describes data validation in Input Pro controls. After a general overview of data validation, the paper provides detailed information about data validation in each of the Input Pro controls that performs data validation: fpCurrency, fpDateTime, fpDoubleSingle, fpLongInteger, fpMask, fpMemo, and fpText.

*Data validation* is the process by which the control determines if supplied data is valid. All the formatted edit controls and the fpMemo control provide data validation.

You can customize how the control defines invalid data and when the control validates the data. In addition, you can turn off data validation.

---

## What Is Valid Data?

---

Each control provides default settings that define valid data for the control. Each control also provides additional properties to let you change the default settings or define additional types of invalid data. For example, the fpCurrency control has predefined minimum and maximum values, which you can change if you prefer.

For a list of the default settings and available properties for defining valid data for each control, see the section listed in the following table:

<b>Control</b>	<b>Valid Data discussion on</b>
fpCurrency	page 5
fpDateTime	page 7
fpDoubleSingle	page 11
fpLongInteger	page 13
fpMask	page 16
fpMemo	page 18
fpText	page 20

---

## When Is Data Validated?

---

All the formatted edit controls can validate data when the data is being entered in the control and when the control loses the focus.

The formatted edit controls consist of the fpCurrency, fpDateTime, fpDoubleSingle, fpLongInteger, fpMask, and fpText controls. These controls are formatted edit controls because they can validate data according to a provided set of valid data, and they can format

the data in the control. The validation and formatting for these controls for data entered by users are controlled by the **UserEntry** property, which for all controls except the fpDateTime control, is set by default to 0 (Formatted).

The following sections describe in detail what happens

- as data comes into the control
- when the control loses the focus

---

### *As Data Comes into the Control*

Data comes into the control from four sources: the user typing, the user pasting, a database, or code. All the controls differentiate between the sources of the data when the data comes into the control as follows:

- For all the formatted edit controls except the fpMask control, the control validates the data as the user enters the data, either by typing or pasting it into the control. The control does not validate data from code or a database as it comes into the control; it validates such data only when the control loses the focus.
- The fpMask control and the fpMemo control validate all data as it is entered into the control, no matter the source of the data.

Besides the preceding methods, the fpDateTime control also allows users to input a date from a pop-up or drop-down calendar or clock, and the formatted edit controls can let users change data in the control using buttons.

### *What Does the Control Do as Data Comes into the Control?*

All the controls that provide data validation can check data as the user types it or pastes it into the control. The control checks the data against its criteria for valid data.

The following table provides a summary of how the controls behave:

<b>Control</b>	<b>Behavior</b>
fpCurrency, fpDoubleSingle, fpLongInteger, fpText	These controls by default check data as the user types it or pastes it into the control. They cannot check data as it is provided from a database or code.
fpDateTime	The fpDateTime control can check data as the user types it or pastes it into the control, but this control does not do so by default. It cannot check data as it is provided from a database or code.
fpMask and fpMemo	The fpMask and fpMemo controls by default validate not only data typed or pasted by the user as it is entered, but also data provided from code or a database as it comes into the control.

If you prefer that a formatted edit control not validate the data as it is typed or pasted by the user and instead accept any data provided, set the **UserEntry** property for the control to 1 (Free Format). For the fpMask control, you can prevent the control from validating data provided from code or a database by setting the **BestFit** property to True. You cannot turn off data validation for the fpMemo control.

*How Does the Control Respond to Invalid Data?*

While the user is entering data, if the control is validating the data being entered, the control responds to invalid data by blocking the data and firing the **UserError** event. You can respond to the **UserError** event to provide information to the user about what valid data is for this control.

The control can also sound a beep when the user enters invalid data. You can turn on the ability to sound a beep by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If a formatted edit control provides buttons for users to change the value in the control, the buttons can stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

When the control is validating the data as the user enters it, there are certain situations when invalid data can be typed or pasted into the control, as described in the sections for the *fpCurrency*, *fpDateTime*, *fpDoubleSingle*, and *fpLongInteger* controls.

---

*When the Control Loses the Focus*

By default, all the formatted edit controls and the *fpMemo* control validate the data in the control when the control loses the focus, no matter the source of the data. The control checks the data against its criteria for valid data.

When a control receives data from code or from a database and does not currently have the focus, the control does not receive and then lose the focus when the data is placed into the control. Instead, if the control does not have the focus when it receives the data, the control validates the data when the data is placed into the control, similar to when the control validates the data as it loses the focus after the user types it. Throughout this paper, references to the control's behavior when it "loses the focus" are also intended to describe the control's behavior when it receives data from code or a database and does not have the focus.

*What Does the Control Do when It Loses the Focus?*

When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

For example, the user might have failed to enter all the required data into an *fpMask* control that has a mask defined to require that the control be filled when it loses the focus.

If you want to provide your own validation or formatting upon the control losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False for any of the formatted edit controls.

**Note** You cannot prevent the *fpMemo* control from validating the data in the control when it loses the focus.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic's **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

*What Happens to Data when the Control Loses the Focus?*

If the data in the control is invalid, the data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control by setting the **InvalidOption** property.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

---

## fpCurrency Control Data Validation

---

The following sections describe valid data for the fpCurrency control and how the fpCurrency control provides data validation.

*What is Valid Data for the fpCurrency Control?*

Valid characters for the fpCurrency control are numbers, a currency symbol, and a decimal symbol (though the control can supply additional formatting characters, such as a separator character).

The control has a minimum and maximum allowed value. By default, the minimum allowed value is -9,000,000,000 and the maximum allowed value is 9,000,000,000. You can change these values by setting the **MinValue** and **MaxValue** properties.

By default, the control does not accept the Null value, and when a Null value is entered, the control displays its default value, "\$0.00", instead. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

*When is Data Validated for the fpCurrency Control?*

The fpCurrency control by default validates data as the user types or pastes it into the control and when the control loses the focus.

*As Data Comes into the fpCurrency Control**What Does the fpCurrency Control Do as Data Comes into the Control?*

By default, the fpCurrency control validates data as the user types or pastes data into the control. The fpCurrency control does not validate data as it is provided from a database or from code.

If you prefer, the control will not validate data as the user types or pastes it into the control. To turn off data validation as the user types or pastes data, set the **UserEntry** property to 1 (Free Format).

*How Does the fpCurrency Control Respond to Invalid Data?*

When the user tries to type or paste invalid data into the fpCurrency control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, “\$0.00”.

If the fpCurrency control provides buttons for user input, you can have the buttons stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

When the control is validating the data as the user enters it, if you set a minimum value using the **MinValue** property for the fpCurrency control, the control does not validate for data less than the minimum value as the user types or pastes it into the control. The control has to allow values below the minimum so users can enter values within the range. For example, if the control’s minimum value is set to 10, and the user types in 2, the control does not know if the user is finished typing (the user might be starting to type in 20). However, the control does validate the value against the defined minimum value when the control loses the focus.

*When the  
fpCurrency  
Control Loses  
the Focus*

*What Does the fpCurrency Control Do when It Loses the Focus?*

By default, the fpCurrency control validates the data in the control when the control loses the focus, using the criteria for valid data as described in “What is Valid Data for the fpCurrency Control?” on page 5, no matter what the source of the data. When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

You can define the format for the data in the fpCurrency control using the following properties:

CurrencyDecimalPlaces	CurrencyNegFormat	CurrencyPlacement
CurrencySymbol	DecimalPoint	FixedPoint
LeadZero	Separator	UseSeparator

If you want to provide your own validation or formatting upon losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic’s **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

*What Happens to Data when the fpCurrency Control Loses the Focus?*

If the data in the control is invalid, the data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control, by setting the **InvalidOption** property.

If the control contains an invalid value and the **InvalidOption** property is set to 2 (Clear Data), the control displays its default value, “\$0.00”, if the **AllowNull** property is False, or the control is set to the Null value if the **AllowNull** property is True.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

### *fpCurrency Data Validation Summary*

The fpCurrency control validates data as shown in the following table. This table shows the default settings for data validation for the control. You can change these behaviors as described in the preceding sections.

#### **fpCurrency Control Default Data Validation**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
User typing	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
User pasting	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Database	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Code	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, “\$0.00”	Not applicable	

## fpDateTime Control Data Validation

The following sections describe valid data for the fpDateTime control and how the fpDateTime control provides data validation.

### *What Is Valid Data for the fpDateTime control?*

The fpDateTime control accepts only numbers and letters that specify a valid date or time, accompanied by valid separator characters. The user can type any of the following characters to indicate distinct data, such as to separate month values from day values: comma (,), hyphen (-), slash (/), period (.), colon (:), space, and backslash (\). Regardless of which sep-

arator character the user types, the fpDateTime control displays separator characters according to the defined format.

The characters that the fpDateTime control considers invalid depend on the date/time format. By default, the fpDateTime control uses the short date format specified in the Windows international or regional settings. You can change the format used by the fpDateTime control. For instructions, see the *Input Pro User's Guide*.

The control has minimum and maximum allowed date and time values. By default, the minimum allowed date value is January 1, 100, and the maximum allowed date value is December 31, 9999. You can change these values by setting the **DateMin** and **DateMax** properties. By default, the minimum and maximum allowed time values are determined by the user's system international or regional settings. You can override the system settings by setting the **TimeMin** and **TimeMax** properties.

By default, the control does not accept the Null value, and treats it as invalid data. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

---

### *When Is Data Validated for the fpDateTime control?*

The fpDateTime control by default validates data when the control loses the focus. The fpDateTime control does not check data as it is entered by default, but you can have the control do so.

### *As Data Comes into the fpDateTime Control*

#### *What Does the fpDateTime Control Do as Data Comes into the Control?*

By default, the fpDateTime control does not validate data as the user types or pastes data into the control. If you prefer, the control will validate data as the user types or pastes it into the control. To turn on data validation as the user types or pastes data, set the **UserEntry** property to 0 (Formatted).

The fpDateTime control cannot validate data as it is provided from a database or from code.

#### *How Does the fpDateTime Control Respond to Invalid Data?*

If the control is validating data as the user types or pastes it into the control, when the user tries to type or paste invalid data into the fpDateTime control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, the current date and time.

If the fpDateTime control provides buttons for user input, you can have the buttons stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

Even if you have set the **UserEntry** property to 0 (Formatted) so the control checks data as it is entered, there are times when the fpDateTime control will allow the user to enter invalid data. These situations are:

1. Date values larger than the maximum date and less than the minimum date are allowed in the fpDateTime control.  
Though the invalid value is allowed in the control, the **UserError** event (error code 7) occurs when the invalid value is entered. The control also validates the value when the control loses the focus, at which time the **InvalidData** event occurs.
2. Time values larger than the maximum time and less than the minimum time are allowed in the fpDateTime control.  
Though the invalid value is allowed in the control, the **UserError** event (error code 7) occurs when the invalid value is entered. The control also validates the value when the control loses the focus, at which time the **InvalidData** event occurs.

*When the fpDateTime Control Loses the Focus*

*What Does the fpDateTime Control Do when It Loses the Focus?*

By default, the fpDateTime control validates the data in the control when the control loses the focus, using the criteria for valid data as described in “What Is Valid Data for the fpDateTime control?” on page 7, no matter what the source of the data. When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

You can define the format for the data in the fpDateTime control using the following properties:

DateTimeFormat	LongDayName	LongMonthName
ShortDayName	ShortMonthName	TimeString1159
TimeString2359	TimeStyle	UserDefinedFormat

If you want to provide your own validation or formatting upon losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic’s **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

*What Happens to Data when the fpDateTime Control Loses the Focus?*

If the control contains an incomplete date or time, the control attempts to calculate the intended date or time using the method specified by the **DateCalcMethod** property. The **DateCalcMethod** property uses predefined algorithms or the settings of the **DateDefault**, **TimeDefault**, or **DateCalcY2K** properties to calculate the intended date or time. For more information about the settings for these properties, refer to the *Input Pro Reference Guide*.

If the fpDateTime control cannot calculate the date or time value using the method specified by the DateCalcMethod property, or the data is just invalid (for example, “1/32/99”), the invalid data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control by setting the **InvalidOption** property.

If the control contains an invalid value and the **InvalidOption** property is set to 2 (Clear Data), the control displays its default value, the current date and time, if the **AllowNull** property is False, or the control is set to the Null value if the **AllowNull** property is True.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

### *fpDateTime Data Validation Summary*

The fpDateTime control validates data as shown in the following table. This table shows the default settings for data validation for the control. You can change these behaviors as described in the preceding sections.

**fpDateTime Control Default Data Validation**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
User typing	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs*
User pasting	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs*
Database	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs*
Code	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs*
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, the current date	Not applicable	

\*If the data is incomplete, but not invalid, the control tries to calculate the date using the method specified by the **DateCalcMethod** property.

---

## fpDoubleSingle Control Data Validation

---

The following sections describe valid data for the fpDoubleSingle control and how the fpDoubleSingle control provides data validation.

---

### *What Is Valid Data for the fpDoubleSingle Control?*

Valid characters for the fpDoubleSingle control are numbers and a decimal symbol (though the control can supply additional formatting characters, such as a separator character).

The control has a minimum and maximum allowed value. By default, the minimum allowed value is -9,000,000,000 and the maximum allowed value is 9,000,000,000. You can change these values by setting the **MinValue** and **MaxValue** properties.

By default, the control does not accept the Null value, and treats it as invalid data. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

---

### *When Is Data Validated in the fpDoubleSingle Control?*

The fpDoubleSingle control by default validates data as the user types or pastes it into the control and when the control loses the focus.

---

### *As Data Comes into the fpDoubleSingle Control*

#### *What Does the fpDoubleSingle Control Do as Data Comes into the Control?*

By default, the fpDoubleSingle control validates data as the user types or pastes data into the control. The fpDoubleSingle control does not validate data as it is provided from a database or from code.

If you prefer, the control will not validate data as the user types or pastes it into the control. To turn off data validation as the user types or pastes data, set the **UserEntry** property to 1 (Free Format).

#### *How Does the fpDoubleSingle Control Respond to Invalid Data?*

When the user tries to type or paste invalid data into the fpDoubleSingle control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, "0".

If the fpDoubleSingle control provides buttons for user input, you can have the buttons stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

When the control is validating the data as the user enters it, if you set a minimum value using the **MinValue** property for the fpDoubleSingle control, the control does not validate for data less than the minimum value as the user types or pastes it into the control. The control has to allow values below the minimum so users can enter values within the range. For example, if the control's minimum value is set to 10, and the user types in 2, the control does not know if the user is finished typing (the user might be starting to type in 20). However, the control does validate the value against the defined minimum value when the control loses the focus.

*When the fpDoubleSingle Control Loses the Focus*

*What Does the fpDoubleSingle Control Do when It Loses the Focus?*

By default, the fpDoubleSingle control validates the data in the control when the control loses the focus, using the criteria for valid data as described in “What Is Valid Data for the fpDoubleSingle Control?” on page 11, no matter what the source of the data. When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

You can define the format for the data in the fpDoubleSingle control using the following properties:

DecimalPlaces	DecimalPoint	FixedPoint
LeadZero	NegFormat	Separator
UseSeparator		

If you want to provide your own validation or formatting upon losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic's **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

*What Happens to Data when the fpDoubleSingle Control Loses the Focus?*

If the data in the control is invalid, the data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control by setting the **InvalidOption** property.

If the control contains an invalid value and the **InvalidOption** property is set to 2 (Clear Data), the control displays its default value, “0”, if the **AllowNull** property is False, or the control is set to the Null value if the **AllowNull** property is True.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

---

### *fpDouble- Single Data Validation Summary*

The fpDoubleSingle control validates data as shown in the following table. This table shows the default settings for data validation for the control. You can change these behaviors as described in the preceding sections.

#### fpDoubleSingle Control Default Data Validation

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
User typing	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
User pasting	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Database	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Code	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, "0"	Not applicable	

---

## fpLongInteger Control Data Validation

---

The following sections describe valid data for the fpLongInteger control and how the fpLongInteger control provides data validation.

---

### *What Is Valid Data for the fpLong- Integer Control?*

Valid characters for the fpLongInteger control are numbers (though the control can supply additional formatting characters, such as a separator character).

The control has a minimum and maximum allowed value. By default, the minimum allowed value is -2,147,483,648 and the maximum allowed value is 2,147,483,647. You can change these values by setting the **MinValue** and **MaxValue** properties.

By default, the control does not accept the Null value, and treats it as invalid data. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

---

### *When Is Data Validated in the fpLongInteger Control?*

The fpLongInteger control by default validates data as the user types or pastes it into the control and when the control loses the focus.

### *As Data Comes into the fpLongInteger Control*

#### *What Does the fpLongInteger Control Do as Data Comes into the Control?*

By default, the fpLongInteger control validates data as the user types or pastes data into the control. The fpLongInteger control does not validate data as it is provided from a database or from code.

If you prefer, the control will not validate data as the user types or pastes it into the control. To turn off data validation as the user types or pastes data, set the **UserEntry** property to 1 (Free Format).

#### *How Does the fpLongInteger Control Respond to Invalid Data?*

When the user tries to type or paste invalid data into the fpLongInteger control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, "0".

If the fpLongInteger control provides buttons for user input, you can have the buttons stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

When the control is validating the data as the user enters it, if you set a minimum value using the **MinValue** property for the fpLongInteger control, the control does not validate for data less than the minimum value as the user types or pastes it into the control. The control has to allow values below the minimum so users can enter values within the range. For example, if the control's minimum value is set to 10, and the user types in 2, the control does not know if the user is finished typing (the user might be starting to type in 20). However, the control does validate the value against the defined minimum value when the control loses the focus.

### *When the fpLongInteger Control Loses the Focus*

#### *What Does the fpLongInteger Control Do when It Loses the Focus?*

By default, the fpLongInteger control validates the data in the control when the control loses the focus, using the criteria for valid data as described in "What Is Valid Data for the fpLong-Integer Control?" on page 13, no matter what the source of the data. When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

You can define the format for the data in the fpLongInteger control using the following properties:

NegFormat                      Separator                      UseSeparator

If you want to provide your own validation or formatting upon losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic's **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

#### *What Happens to Data when the fpLongInteger Control Loses the Focus?*

If the data in the control is invalid, the data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control by setting the **InvalidOption** property.

If the control contains an invalid value and the **InvalidOption** property is set to 2 (Clear Data), the control displays its default value, "0", if the **AllowNull** property is False, or the control is set to the Null value if the **AllowNull** property is True.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

### *fpLongInteger Data Validation Summary*

The fpLongInteger control validates data as shown in the following table. This table shows the default settings for data validation for the control. You can change these behaviors as described in the preceding sections.

#### **fpLongInteger Control Default Data Validation**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
User typing	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
User pasting	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs

**fpLongInteger Control Default Data Validation (Continued)**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
Database	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Code	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, "0"	Not applicable	

**fpMask Control Data Validation**

The following sections describe valid data for the fpMask control and how the fpMask control provides data validation.

***What Is Valid Data for the fpMask Control?***

Valid values for the fpMask control can contain letters, numbers, and symbols. For a mask created with the **Mask** property, which characters are considered invalid depends on the position of the character and the definition of its corresponding mask character. If any character conflicts with the mask, the value is considered invalid. For example, with a telephone number mask such as "(###) ###-####", all characters other than numbers are invalid.

By default, the control does not accept the Null value, and treats it as invalid data. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

***When Is Data Validated in the fpMask Control?***

The fpMask control by default validates data as the user types or pastes it into the control, as data is entered from a database or by code, and when the control loses the focus.

***As Data Comes into the fpMask Control******What Does the fpMask Control Do as Data Comes into the Control?***

The fpMask control validates not only data typed or pasted by the user as it is entered, but also data provided from code or a database as it comes into the control. If you prefer that the control does not check the data as it is typed or pasted by the user and instead accepts any data provided, you can set the **UserEntry** property for the control to 1 (Free Format). If you prefer that the control does not check the data as it is provided from code or a database and instead accepts any data provided, you can set the **BestFit** property to True.

*How Does the fpMask Control Respond to Invalid Data?*

When the user tries to type or paste invalid data into the fpMask control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, an empty string.

If the fpMask control provides buttons for user input, you can have the buttons stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

*When the  
fpMask Control  
Loses the  
Focus*

*What Does the fpMask Control Do when It Loses the Focus?*

By default, the fpMask control validates the data in the control when the control loses the focus, using the criteria for valid data as described in “What Is Valid Data for the fpMask Control?” on page 16, no matter what the source of the data. When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

You can define the format for the data in the fpMask control using the **Mask** property.

If you want to provide your own validation or formatting upon losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic’s **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

*What Happens to Data when the fpMask Control Loses the Focus?*

If the data in the control is invalid, the data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control by setting the **InvalidOption** property.

If the control contains an invalid value and the **InvalidOption** property is set to 2 (Clear Data), the control displays its default value, an empty string, when it loses the focus.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

---

### *fpMask Data Validation Summary*

The fpMask control validates data as shown in the following table. This table shows the default settings for data validation for the control. You can change these behaviors as described in the preceding sections.

#### **fpMask Control Default Data Validation**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
User typing	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
User pasting	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Database	Yes	Allowed in the control (up to the size of the defined mask); <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Code	Yes	Allowed in the control (up to the size of the defined mask); <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, an empty string	Not applicable	

---

## **fpMemo Control Data Validation**

---

The following sections describe valid data for the fpMemo control and how the fpMemo control provides data validation.

---

### *What Is Valid Data for the fpMemo Control?*

The fpMemo control accepts any character or value into the control, except the Null value when the **AllowNull** property is set to False.

By default, the control does not accept the Null value, and treats it as invalid data. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

The fpMemo control has a line limit. Depending on the method by which data is provided for the control, the control either blocks data or truncates data that is beyond the specified line limit.

### When Is Data Validated in the fpMemo Control?

The fpMemo control by default validates data as the user types it into the control, and as data is entered from a database or by code. The fpMemo control also validates data when the control loses the focus.

If the number of lines of text in the control exceed the line limit, the control blocks the user from entering more data, and the **UserError** event occurs. If the data was provided by pasting, the control allows the data at the time the data is pasted into the control, but when the control loses the focus, it truncates the data to the number of lines allowed in the control.

### As Data Comes into the fpMemo Control

#### What Does the fpMemo Control Do as Data Comes into the Control?

The fpMemo control validates data as the user types it into the control. The fpMemo control also validates data as it is provided from a database or from code.

#### How Does the fpMemo Control Respond to Invalid Data?

When the user tries to type data that is longer than the number of lines allowed in the fpMemo control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, an empty string.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

### fpMemo Data Validation Summary

The fpMemo control validates data as shown in the following table. This table shows the default settings for data validation for the control.

#### fpMemo Control Default Data Validation

Data Source	Validated at Entry	Invalid Data is . . .	Validated when Control Loses the Focus	Invalid Data is . . .
User typing	Yes	Blocked by the control; <b>UserError</b> event occurs	Not applicable	
User pasting	No	Allowed in the control	Yes	Truncated to the number of lines allowed by the <b>LineLimit</b> property

**fpMemo Control Default Data Validation (Continued)**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
Database	Yes	Truncated to the number of lines allowed by the <b>LineLimit</b> property	Not applicable	
Code	Yes	Truncated to the number of lines allowed by the <b>LineLimit</b> property	Not applicable	
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, an empty string	Not applicable	

---

**fpText Control Data Validation**

---

The following sections describe valid data for the fpText control and how the fpText control provides data validation.

---

***What Is Valid Data for the fpText Control?***

The fpText control accepts all characters, including numbers, letters, and symbols. The fpText control handles numbers, letters, and symbols as character data. You can restrict the valid characters by setting the **CharValidationText** property.

For example, if you set the **CharValidationText** property to "ABCD", the user can type only those four characters, but both of the strings "AABBCADDB" and "BBCC" are valid. The **CharValidationText** property is case sensitive. Unlike the fpMask control, the fpText control does not require that the valid characters be in a certain position.

By default, the control does not accept the Null value, and treats it as invalid data. You can allow the Null value in the control as valid data by setting the **AllowNull** property to True.

---

***When Is Data Validated in the fpText Control?***

The fpText control by default validates data as the user types or pastes it into the control and when the control loses the focus.

---

***As Data Comes into the fpText Control******What Does the fpText Control Do as Data Comes into the Control?***

By default, the fpText control validates data as the user types or pastes data into the control. The fpText control does not validate data as it is provided from a database or from code.

If you prefer, the control will not validate data as the user types or pastes it into the control. To turn off data validation as the user types or pastes data, set the **UserEntry** property to 1 (Free Format).

#### *How Does the fpText Control Respond to Invalid Data?*

When the user tries to type or paste invalid data into the fpText control, the control blocks the data and fires the **UserError** event. The control can also sound a beep when the user enters invalid data, which you can turn on by setting the **AutoBeep** property to True or by setting the *AutoBeep* parameter in the **UserError** event to True.

If the user tries to set the value to the Null value, and the **AllowNull** property is set to False, the control displays its default value, "0".

If the fpText control provides buttons for user input, you can have the buttons stop changing the value to prevent invalid data (for example, the control can stop incrementing or decrementing the value in response to the buttons). To use the buttons to prevent invalid data, set the **ButtonIndex**, **ButtonMin**, **ButtonMax**, and **ButtonWrap** properties.

#### *When the fpText Control Loses the Focus*

##### *What Does the fpText Control Do when It Loses the Focus?*

By default, the fpText control validates the data in the control when the control loses the focus, using the criteria for valid data as described in "What Is Valid Data for the fpText Control?" on page 20, no matter what the source of the data. When the control loses the focus, the control tries to format the data in the control using the format you have defined for the data, and it validates the data. If the data is invalid or the control cannot format the data to the expected format, an **InvalidData** event occurs.

You can define the format for the data in the fpText control using the following properties:

AutoCase                      PasswordChar

If you want to provide your own validation or formatting upon losing focus and do not want the control to attempt to validate and format the data for you, set the **AllowModifiedFlag** property to False.

If you want to provide your own validation and you are using Visual Basic 6.0, you might want to set the **CausesValidation** property for the next control on the form to True and use Visual Basic's **Validate** event to provide your own validation, after turning off the validation provided by Input Pro controls.

##### *What Happens to Data when the fpText Control Loses the Focus?*

If the data in the control is invalid, the data remains in the control and the background color of the control changes to the color specified by the **InvalidColor** property. You can specify that the control does not display invalid data, but instead hides it or clears it from the control by setting the **InvalidOption** property.

If the control contains an invalid value and the **InvalidOption** property is set to 2 (Clear Data), the control displays its default value, an empty string.

**Note** If the control allows the Null value and the data in the control is a Null value, the background color will change to the color set by the **NullColor** property.

### *fpText Data Validation Summary*

The fpText control validates data as shown in the following table. This table shows the default settings for data validation for the control. You can change these behaviors as described in the preceding sections.

#### **fpText Control Default Data Validation**

<b>Data Source</b>	<b>Validated at Entry</b>	<b>Invalid Data is . . .</b>	<b>Validated when Control Loses the Focus</b>	<b>Invalid Data is . . .</b>
User typing	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
User pasting	Yes	Blocked by the control; <b>UserError</b> event occurs	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Database	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Code	No	Allowed in the control	Yes	Displayed with different background color; <b>InvalidData</b> event occurs
Any source when data is Null value	Yes	If <b>AllowNull</b> property is False, Null value is not allowed and control displays default value, an empty string	Not applicable	

## About FarPoint

---

FarPoint Technologies, Inc., a privately held company with corporate headquarters located in Morrisville, North Carolina, is a leading developer and publisher of professional components for Windows development. Our award-winning tools benefit the ' leading corporations, software companies, and independent consultants around the world as a cost-effective solution for building distributed enterprise-wide applications for commercial or in-house use. From the CEO to our newest team member, our goals are the same: to provide you with the innovative tools and support you need to effectively compete in today's competitive development market.

FarPoint's foundation began in 1991 (then Prescription Software) by re-introducing Drivers Professional Toolbox for Windows, what is believed to be the first development package available for the professional Windows developer, with thirteen new DLLs, including a fully featured spreadsheet control. Developers using languages such as C or C++ quickly realized the benefits of using our pre-built DLLs instead of spending their valuable resources developing these controls internally.

The advent of Visual Basic 1.0 presented FarPoint with a new opportunity in the professional development market. After being contacted by Microsoft asking that our spreadsheet be made available for their users, the first spreadsheet control ever available for Visual Basic was soon introduced in our product Visual Architect. Along with the spreadsheet control, Visual Architect offered formatted-edit controls in this new VBX format. The rising popularity of Visual Basic and our controls allowed us to introduce several new products to the market in the coming months: Spread/VBX, Aware/VBX, Grid/VBX, Tab/VBX and Tab Pro, while continuing support for Professional Toolbox for Windows.

Today, FarPoint is one of the oldest and most respected vendors in the industry. We continue to cater our products to both the professional Visual Basic and Visual C++ developer by offering our components as a 32-bit ActiveX control, 16- and 32-bit DLLs and a VBX control.

Whether you are developing applications for COM or .NET, FarPoint has a component or control that is right for you and that will deploy royalty-free in the environment you need.

For more information, check out our web site or contact us directly at the North American office or European office of FarPoint:

### North America Contact

FarPoint Technologies, Inc.  
808 Aviation Parkway  
Suite 1300  
Morrisville, NC 27560 USA

Phone: 919-460-4551  
email: [fpsales@fpoint.com](mailto:fpsales@fpoint.com)  
Web: [www.fpoint.com](http://www.fpoint.com)

### Europe Contact

FarPoint Europe Ltd.  
Whiteleaf, Roundabout Lane  
West Chilmington  
Pulborough, West Sussex RH20 2RL  
England

Tele: +44 (0) 1798 812 372  
Fax: +44 (0) 1798 813 049  
email: [salesEurope@fpoint.com](mailto:salesEurope@fpoint.com)

Information in the white paper is subject to change without notice and does not represent a commitment on the part of FarPoint Technologies, Inc.

© 1999-2006 FarPoint Technologies, Inc. All rights reserved.

Unless otherwise noted, all names of companies, products, street addresses, and persons contained herein are part of a completely fictitious scenario or scenarios and are designed solely to document the use of a FarPoint Technologies, Inc., product.

Input Pro is a trademark of FarPoint Technologies, Inc. ActiveX, Microsoft, and Word for Windows are either registered trademarks or trademarks of Microsoft Corporation.