

Re: One check box or the other

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.formscoding/2006-08/msg00244.html>

- *From:* Klatuu <Klatuu@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Thu, 20 Jul 2006 13:30:01 -0700
-

Thanks for the kudos, BruceM. I appreciate your hanging in there. Maybe I can clear it up for you.

First, you would need the two check boxes bound to the two controls, Requested and Received. This will give you access to these data elements so they will update with the record. Now, so as not to confuse the user, we want to hide them. To do that, set these properties for both check boxes:

Visible = No
Tab Stop = No
Default Value = False

Now the user will not see the controls and tabbing through the form will skip them.

Then you need the option group with 2 option buttons. For the Option Group, set the Default Value to Null. Set the Option Value for Required to 0 and the Option Value for Received to 1.

Put this code (with names changed where necessary) in the After Update event of the option group. If you do not click on either option button, both check boxes will remain False. When you click on the Requested option button, the Requested check box will be changed to True and the Received Check box will stay False. When you click on the received option button, the Requested check box = change to False and the Received check box will change to True.

```
Select Case Me.opgTripleChoice
Case is Null
Me.chkOne = False
Me.chkTwo = False
Case is 0
Me.chkOne = True
Me.chkTwo = False
Case is 1
Me.chkOne = False
Me.chkTwo = True
End Select
```

Then this code goes in the Form's Current event. Since the Option Group is not bound to any field in the table, it will not be affected by moving to

Re: One check box or the other

different record or inserting a new record. The check boxes will, so when we get a change in current record, the Current event fires and makes the option group reflect what is in the record:

```
If Me.NewRecord Then 'We want to start with null
Me.opgTripleChoice = Null
ElseIf Me.chkRequested = True And Me.chkReceived = False Then
Me.optTripeChoice = 0
ElseIf Me.chkRequested = False And Me.chkReceived = True Then
Me.optTripeChoice = 1
Else
Me.opgTripleChoice = Null
End If
```

Now, there is another way you could do this. Rather than having two boolean fields, one for requested and one for received, use one interger field and bind it to the option group. Use the value in that field to determine the status of requested or received or neither.

"BruceM" wrote:

I often read your posts, and have a lot of respect for what you have to offer in this group. Please realize that I looked at your code for quite some time, but could not figure out what I was supposed to do. It isn't that I am ignoring what you said, but rather that I was not able to understand it. If I was ignoring it I would either have not responded (not my style) or I would have thanked you for the suggestion and left it at that (also not my way of doing things). Instead I tried to understand, and asked questions where I could not.

After reading your most recent post, I think I understand that you are suggesting I use an option group to control the hidden check boxes. By the way, it wouldn't be a big deal to switch to an option group, and forget about the check boxes. In either case I could have a two-item option group when needed, and a three-item group the rest of the time.

"Klatuu" <Klatuu@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message news:3EFDE551-12C8-48A7-92B9-84C98B944D11@xxxxxxxxxxxxxxxxxxxx

Yes, you are missing the point. If you review the code I posted, it provides

exactly what you are stating.

If neither option is selected, both fields in your table will be False (unchecked)

If option 1 is selected, Requested will be True and Received will be False

If option 2 is selected, Requested will be False and Received will be True.

You do have need for a third option. You options, as you are wanting is

Re: One check box or the other

what the functionality of the Triple State Option Group provides.

I don't see the controls as redundant. I see them as providing the functionality you require. If you choose to use only the bound check boxes, then you will have to write code in the after update event of each to evaluate the other and determine whether it is valid or not. Or you could use the form's Before Update event.

Using hidden controls as the bound controls is a very accepted and usual way to do things. Think of using a Combo to look up a specific record. Because of the way a combo works, it becomes problematic to use it as a bound control in this instance. So, we use a text box, but since the user can already see the value in the combo, why show it to them twice?

If you choose not to use the suggestion I provided, that's fine, but I really do think it is the easiest to implement and control.

"BruceM" wrote:

Thanks for taking the time to look at this and to reply, but I'm not following your suggestion. First, I should have said that both boxes can't be checked, but in some cases both can be unchecked. When a vendor's certificate is due to expire we request an updated certificate, and check a Requested box. After the new certificate arrives (or when we receive a certificate from a new vendor) we check the Received box. One or the other of these is always checked. The certificate typically expires three years from the issue date. A year (or two years) before the expiration date we need to get confirmation of status. For this there is another pair of Requested/Received check boxes. For a new vendor we do not request confirmation until they have been a vendor for at least a year, so these check boxes are both blank until the

Re: One check box or the other

first time we request confirmation.
I don't really have a use for the third category, although I suppose it could be New Vendor or something of the sort. But I have to be honest that I don't see the value of using an option group to control a pair of check boxes. In effect there would be redundant controls on the form. Or am I missing something here? One part in particular that I don't understand is "you can control them using a hidden check box for each field that is the bound control."

"Klatuu" <Klatuu@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message news:3EA78EED-B135-4FF6-831F-06682EF84A53@xxxxxxxxxxxxxxxxxxxx

Triple state makes perfect sense for this. You say that they cannot be the same, but in some cases neither would be selected. Well, those two statements appear to be contradictory, because neither selected would make them both unchecked. So, to have one or the other (but not both) checked, an option group is exactly what you want. Now, you also want it so that neither is checked. This is the Tripe State option where the value of the option group is Null.

If the current situation is that you have two boolean fields bound to the current two check boxes, you can control them using a hidden check box for each field that is the bound control. The option group would not be bound, but you can use its After Update event to set the values of the hidden check

Re: One check box or the other

boxes.

```
Select Case Me.opgTripleChoice
Case is Null
Me.chkOne = False
Me.chkTwo = False
Case is 0
Me.chkOne = True
Me.chkTwo = False
Case is 1
Me.chkOne = False
Me.chkTwo = True
End Select
```

You will also want to ensure the option group is Null for new records and is correctly set for existing records, so add this to the form's Current event:

```
If Me.NewRecord Then
Me.opgTripleChoice = Null
ElseIf Me.chkOne = True And Me.chkTwo = False Then
Me.optTripeChoice = 0
ElseIf Me.chkOne = False And Me.chkTwo = True Then
Me.optTripeChoice = 1
Else
Me.opgTripleChoice = Null
End If
```

"BruceM" wrote:

Triple-state doesn't make sense in this situation, but could come in handy some day. Thanks for the thought.

Do you see a way of simplifying and streamlining the coding? Again, it's not a big deal, but I expect that anything I can learn for

Re: One check box or the other

this
situation
will prove useful in the
future.

"Rick B" <Anonymous>
wrote in message
news:ORjhg6ArGHA.2232@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Just FYI,
you can do
what you
are asking
but, I have
had a case
in
the
past where I
used a
triple-state
checkbox.
Perhaps this
will suit
your
needs.

You change
the field to
a number
field and it
stores "-1"
for a
checked
value, "0"
for an
unchecked
value, and
Null for a
grayed out
checkbox.

Again, this
may or may
not make
sense for
your
particular
application,
but
it worked
very well

Re: One check box or the other

for us where
we had an
activity that
was either
performed,
not yet
performed
and
required, or
not yet
performed
and
optional.

--
Rick B

"BruceM"
<bamoob@xxxxxxxxxxxxxxxxxxxx>
wrote in
message
news:%23L40gxArGHA.1368@xxxxxxxxxxxxxxxxxxxxxxxx

I
have
a
form
on
which
are
two
check
boxes
(bound
to
Yes/No
fields)
that
cannot
both
be
the
same
value.
Under
some
circumstances
neither

Re: One check box or the other

will
be
checked,
so
I
don't
think
I
can
use
an
option
group
unless
I
add
a
third
option,
which
I
don't
want
to
do.
I
know
I
can
add
this
to
one
check
box
(and
the
inverse
to
the
other):

```
Private  
Sub  
Check1_AfterUpdate()
```

```
Me.Check2  
=  
Not  
Me.Check1
```

Re: One check box or the other

End
Sub

However,
I
would
like
to
know
if
I
could
handle
both
check
boxes
from
a
single
function
(or
sub),
or
otherwise
streamline
the
process.
The
code
above,
if
placed
in
a
function
and
called
from
the
After
Update
event,
affects
one
check
box
differently
than
the
other.
I

Re: One check box or the other

Re: One check box or the other

could
write
an
If
statement
to
take
care
of
both
check
boxes,
but
I
wouldn't
be
saving
myself
much
coding.

My
actual
situation
involves
several
pairs
of
check
boxes,
and
other
cases
in
which
a
similar
principle
applies.
It
comes
up
often
enough
that
I
would
like
to
know
if

Re: One check box or the other

Re: One check box or the other

there
is
a
way
to
handle
it
more
efficiently.