Tipps und Tricks zum FastReport 6

gesammelt von Urs Pfister

Stand 15.01.2016

Inhaltsverzeichnis:

Export 4
Feld: Zwei Dienste addieren (Format Tag) 4
Feld: Zwei Dienste addieren (Format Std) 4
Feld: Einzelnes Feld mit Faktor multiplizieren: 4
Feld: Mehrere addierte Felder mit Faktor multiplizieren: 4
Feld: Ein, mit Faktor multipliziertem Feld, zu zwei Felder addieren: 4
Feld: Ein Feld dividieren und runden: 4
Feld: Bedingung 1 5
Feld: Bedingung 2 5
Feld: im Code Bedingung 1 (if – then - else) 5
Feld: im Code Bedingung 2 (if – then - else) 5
Feld: im Code Bedingung 3 (if – then - else) 5
Feld:im Code Bedingung 4 (case of) 6
Child: im Code Bedingung 1 (case of) 6
Feld: im Code Behandlungstage zählen 6
Feld: im Code Ausgabe mit Zeilenumbruch 7
Feld: Dienst Ikon einbinden 7
Feld: Passfoto aus Patientendetail einbinden 8
Feld: Terminartfarbe einbinden 8
Feld: mehrere Feldinhalte verketten 9
Formatierungen: Stunden, Minuten 9
Feldinhalt in Sekunden → Ausgabe: Stunden dezimal 9
Feldinhalt in Sekunden → Ausgabe: Stunden Minuten 9
Feldinhalt in Sekunden → Ausgabe: Stunden Minuten Sekunden 9
Formatierungen: Dezimalwerte 9
Formatierungen: Höhe der Felder optimieren 10
Formatierungen: Zeilen im Zebralook 10
Zebralook einfach 10
Zebralook kompliziert 10
Formatierungen: ein Teil des Textes innerhalb eines Text-Objektes formatieren. 11
Filter: nur Fälle innerhalb der Auswertungsperiode anzeigen 11
Filter: nur rein nummerische Personalnummern 12
Filter: Dienst Beginn und Ende 12
Filter: Bei der Auswertung mehrerer Dienste in Spalten, nur MA mit Werten zeigen 12
Filter: nur Fälle ohne FID 12
Filter: nur der erste Termin des Falles 12
Filter: nach DIS-Stationen in Patienten-Termin Abfragen 12
Filter: Belegungen und leere Betten zum aktuellen Zeitpunkt 12
Filter: keine gelöschte Termine 13
Filter: keine gelöschte Termine in Standard-Query aufheben 13
Bedingung im Code (if – then – else) für Filter: mit oder ohne Visum 13
Bedingung im Code (case): je nach Terminart unterschiedliche Schriftfarbe 14

Bedingung im Code (stringlist): bestimmter Raum aus String entfernen 14
Header für weitere Seiten 15
Zusätzliche Felder PEP 16
PEP Personaldaten – Betrieb – Polypoint / RAP 16
Kostenstelle aus Hierarchie-Gruppe 16
Guthaben Bezug vom 1.1. bis Ende der Auswertungsperiode (PA-Code 501) 16
Zusätzliche Felder RAP 17
Patient Name Alphabet 17
Beginn und Ende mit Wochentag und Zeit (inkl.Termine über Mitternacht) 17
Übersetzung Terminart Bezeichnung 17
Übersetzung Raum und Gerät Bezeichnung 17
DIS Urlaub in Query „Alle Betten“ 17
Einzelne Termin Leistungsmarke ausgeben 17
Terminart Farbe als RGB Code ausgeben 17
Verknüpfung von zwei unterschiedlichen Abfragen: 17
Einfache Verknüpfung (Fall) 17
Zweifache Verknüpfung (Mitarbeiter; PEP-Planungsdatum/RAP-Termindatum) 18
Verknüpfung von zwei nicht-standard Abfragen: 18
Kreuztabelle formatieren: 18
Dialogfenster nicht anzeigen bei nur einer Export Seite 20
Dialogfenster beim Export anzeigen 20
Dialogfenster beim Klick auf „Abbrechen“ schliessen 21
Dialog mit Eingabefeld mit Maske 21
Dialog für mehrere Reportseiten 22
Variante I 22
Variante II 23

Export

1. neue Seite erstellen
2. Seite umbenennen in _EXP_
3. Ausrichtung und Seitenränder an „PageMain“ anpassen.
4. „Header“ und „MasterData“ von „PageMain“ kopieren und in „_EXP_“ einfügen.
5. Im „Header“ (Kontextmenü) Option „Auf neuer Seite wiederholen“ deaktivieren.

Hinweis: Daten aus DOC-Druckvorlagen können so nicht exportiert werden. Allerdings kann der Report im DocAdmin als *.fr3 Datei gespeichert werden, im PEP/RAP FastReport-Editor geöffnet und als Polypoint-Liste abgelegt werden.

Feld: Zwei Dienste addieren (Format Tag)

```
[FloatToDayDecimal
  ( <StatistikM_QRYno3006."Dienste_Tage_[111] Dienst A">
   + <StatistikM_QRYno3006."Dienste_Tage_[112] Dienst B">
   ,True)]
```

Feld: Zwei Dienste addieren (Format Std)

```
[SecondsToHoursDecimals
  ( <StatistikM_QRYno3006."Dienste_Stunden_[111] Dienst A">
   + <StatistikM_QRYno3006."Dienste_Stunden_[112] Dienst B">
   ,2,True)]
```

Feld: Einzelnes Feld mit Faktor multiplizieren:

```
[StrToFloat
  (FloatToDayDecimal
  (<StatistikMK_QRYno3007."Dienste_Tage_[701] Pikett 1">,False)
  )
*1.25]
```

Feld: Mehrere addierte Felder mit Faktor multiplizieren:

```
[StrToFloat
  (FloatToDayDecimal
  ( <StatistikMK_QRYno3007."Dienste_Tage_[701] Pikett 1">
   + <StatistikMK_QRYno3007."Dienste_Tage_[702] Pikett 2">
   + <StatistikMK_QRYno3007."Dienste_Tage_[703] Pikett 3">,False)
  )
*1.25]
```

Feld: Ein, mit Faktor multipliziertem Feld, zu zwei Felder addieren:

```
[
  (StrToFloat
    (FloatToDayDecimal
    ( <StatistikMK_QRYno3007."Dienste_Tage_[701] Pikett 1">,False)
    )
  *1.25)
 +
  (StrToFloat
    (FloatToDayDecimal
    ( <StatistikMK_QRYno3007."Dienste_Tage_[702] Pikett 2">
    + <StatistikMK_QRYno3007."Dienste_Tage_[703] Pikett 3">,False)
    )
  )
]
```

Feld: Ein Feld dividieren und runden:

```
[SecondsToHoursDecimals
  (round
   ((<StatistikM_QRYno3006."Dienste_Stunden_[3050] Militär">)
```

```
    / (<StatistikM_QRYno3006."TAGESSOLLRED">/3600)    )
  ,2,True)
]
```

Feld: Bedingung 1

Wenn keine „Fall_Zimmernummer“ vorhanden dann Text „TK“ eintragen.

```
[Iif(length(<PatientenTermine_QRYno1009."Fall_Zimmernummer">)
>0,<PatientenTermine_QRYno1009."Fall_Zimmernummer">, 'TK')]
```

Feld: Bedingung 2

Wenn in den Patientenangaben das Feld „Land“ leer ist, sollen nur PLZ und Ort ausgegeben werden.

```
[Iif(length(<PatientenTermine_QRYno1009."Patient_Adresse_Land">)= 0,
'',<PatientenTermine_QRYno1009."Patient_Adresse_Land">+' -
')][PatientenTermine_QRYno1009."Patient_Plz"] [PatientenTermine_QRYno1009."Patient_Ort"]
```

Feld: im Code Bedingung 1 (if – then - else)

Wenn Feld „Personal_NameVorname“ „OP“ enthält dann Wert „Belegung_Dauer_Minuten“ sonst Wert „Termin_Dauer_Minuten“

```
procedure RessourcenTermine_QRYno1005Termin_Dauer_MinutenOnBeforePrint(Sender:
TfrxComponent);
var
  memo: TfrxMemoView;
  nPos: Integer;
begin
  memo := TfrxMemoView(Sender);
  nPos := Pos('OP',<RessourcenTermine_QRYno1005."Personal_NameVorname">);
  if (nPos > 0) then
    memo.Text := <RessourcenTermine_QRYno1005."Belegung_Dauer_Minuten">
  else
  begin
    nPos := Pos('IMC',<RessourcenTermine_QRYno1005."Personal_NameVorname">);
    if (nPos > 0) then
      memo.Text := <RessourcenTermine_QRYno1005."Termin_Dauer_Minuten">;
  end;
```

Feld: im Code Bedingung 2 (if – then - else)

Feld „Austritt“ nur anzeigen wenn das Austrittsdatum innerhalb der Auswertungsperiode liegt.

```
procedure Memo8OnBeforePrint(Sender: TfrxComponent);
begin
  if ((<StatistikM_QRYno3006."Personaldaten_Betrieb_Austrittsdatum"> >= <dteLeft>)
   and(<StatistikM_QRYno3006."Personaldaten_Betrieb_Austrittsdatum"> <= <dteRight>))
  then TfrxMemoView(Sender).Text :=
<StatistikM_QRYno3006."Personaldaten_Betrieb_Austrittsdatum">;
end;
```

Feld: im Code Bedingung 3 (if – then - else)

Feld „Fall_Eintritt_Zeit“ nur anzeigen wenn Feld „Fall_Eintritt_Datum“ gleich Feld „Termin_Datum“

```
procedure Memo7OnBeforePrint(Sender: TfrxComponent);
begin
  if (<PatientenTermine_QRYno1009."Fall_Eintritt_Datum"> =
<PatientenTermine_QRYno1009."Termin_Datum">)
  then TfrxMemoView(Sender).Text := <PatientenTermine_QRYno1009."Fall_Eintritt_Zeit">;
end;
```

Feld:im Code Bedingung 4 (case of)

Im des Halter soll NameVornam, Adresse1, Adresse2, PLZ und Ort komagetrennt ausgegeben werden. Wobei die Felder Adresse1 und Adresse2 nicht immer abgefüllt sind. Die Ausgabe darf keine überflüssigen Leerstellen oder Kommas haben.

```
procedure Halter_OnBeforePrint(Sender: TfrxComponent);
var
sHalter, sAdresse, sAdresse1, sAdresse2, sPlzOrt : String;
iLaengeAdr1, iLaengeAdr2 : Integer;
begin
  sHalter := 'Halter: ' + <Stammdaten."Halter_NameVorname"> + ', ';
  sPlzOrt := <Stammdaten."Tier_Halter_Plz"> + ' ' + <Stammdaten."Halter_Ort">;
  sAdresse1 := <Stammdaten."Tier_Halter_Adresse1">;
  sAdresse2 := <Stammdaten."Tier_Halter_Adresse2">;

  if length(sAdresse1)= 0 then iLaengeAdr1 := 0 else iLaengeAdr1 := 1;
  if length(sAdresse2)= 0 then iLaengeAdr2 := 0 else iLaengeAdr2 := 2;
//  ShowMessage(IntToStr(iLaengeAdr1) + ' / ' + IntToStr(iLaengeAdr2));
  case iLaengeAdr1 + iLaengeAdr2 of
       0: sAdresse := '';
       1: sAdresse := sAdresse1 + ', ';
       2: sAdresse := sAdresse2 + ', ';
       3: sAdresse := sAdresse1 + ', ' + sAdresse2 + ', ';
  end;
  TfrxMemoView(Sender).Text := sHalter + sAdresse + sPlzOrt;
end;
```

Child: im Code Bedingung 1 (case of)

Im Child gibt es die Felder Titel, Katalogfeld und Freitext. Der Titel soll nur ausgegeben werden wenn das Katalogfeld und/oder Freitextfeld nicht leer sind.

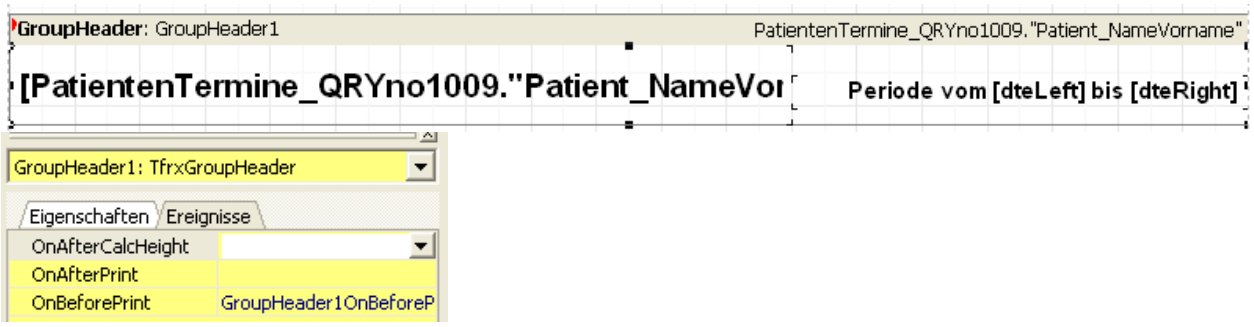
```
procedure Child367OnBeforePrint(Sender: TfrxComponent);
var
iLeer, : integer;    //zum prüfen ob Tab Tumordiagnose leer.
begin
  If engine.FinalPass then // nur wenn DoublePass in den Report Eigenschaften aktiviert
ist
    begin
    if (<ITPA_Biopsien_Tumordiagnose_Version1_1."KATALOG_COMBOBOX1"> ='2696') or
       (<ITPA_Biopsien_Tumordiagnose_Version1_1."KATALOG_COMBOBOX1"> ='') then iLeer :=
iLeer else iLeer := iLeer + 1;
    if (length(<ITPA_Biopsien_Tumordiagnose_Version1_1."TUMOR_1_FREI">) =0) then iLeer :=
iLeer else iLeer := iLeer + 2;
    case iLeer of
         0: child367.visible := false;
         1: begin child367.visible := true;
                  Klg85.visible := true;
                  Memo636.visible := false end;
         2: begin child367.visible := true;
                  Klg85.visible := false;
                  Memo636.visible := true end;
         3: begin child367.visible := true;
                  child367.Height := 50;
                  Klg85.visible := true;
                  Memo636.visible := true;
                  Memo636.Top := 25 end;
    end;
  end;                 // nur wenn DoublePass in den Report Eigenschaften aktiviert ist
end;
```

Feld: im Code Behandlungstage zählen

Variable definieren:

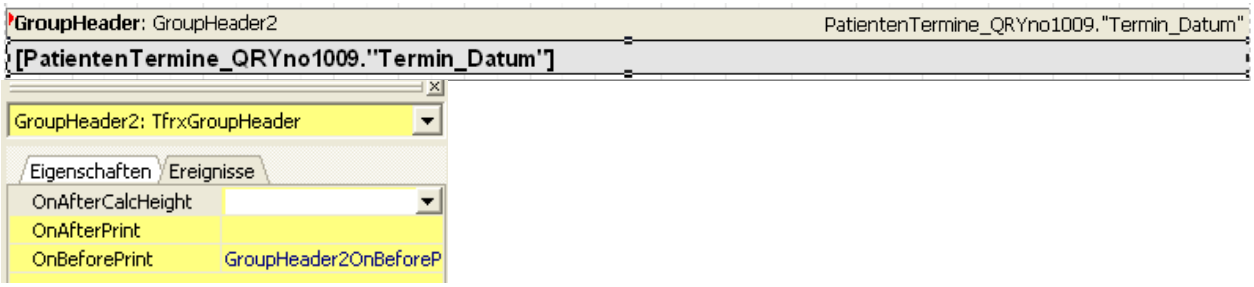
```
var
 iCountDate: Integer;
```

Zähler pro Patient auf null setzen (auf GroupHeader1)



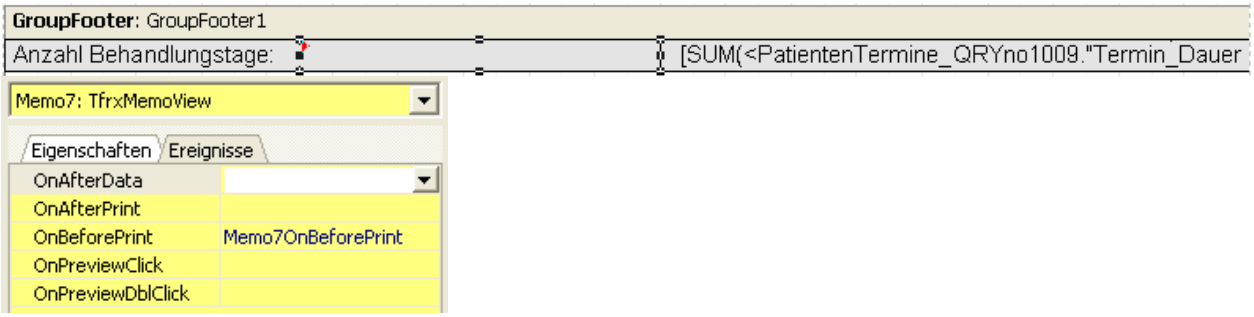

```
procedure GroupHeader1OnBeforePrint(Sender: TfrxComponent);
begin
  iCountDate :=0;
end;
```

Zähler pro Behandlungstag um 1 hochzählen (auf GroupHeader2)




```
procedure GroupHeader2OnBeforePrint(Sender: TfrxComponent);
begin
  iCountDate := iCountDate +1;
end;
```

Total der Behandlungstage im Feld ausgeben (im Memo7)




```
procedure Memo7OnBeforePrint(Sender: TfrxComponent);
begin
  TfrxMemoView(Sender).Text := IntToStr(iCountDate);
end;
```

Feld: im Code Ausgabe mit Zeilenumbruch

```
begin
  if (<RessourcenTermine_QRYno1005."Interne_Termin_Raum_Nummern"> = '63347') or
  (length(<RessourcenTermine_QRYno1005."Interne_Termin_Raum_NummernAlle">) = 0) then
    begin
      TfrxMemoView(Sender).Text := <RessourcenTermine_QRYno1005."Termin_Text_3">;
    end
    else
    begin
      TfrxMemoView(Sender).Text := <RessourcenTermine_QRYno1005."Termin_Raum_NameAlle">
                                 + Chr(13) + <RessourcenTermine_QRYno1005."Termin_Text_3">;
    End;
End;
```

Feld: Dienst Ikon einbinden

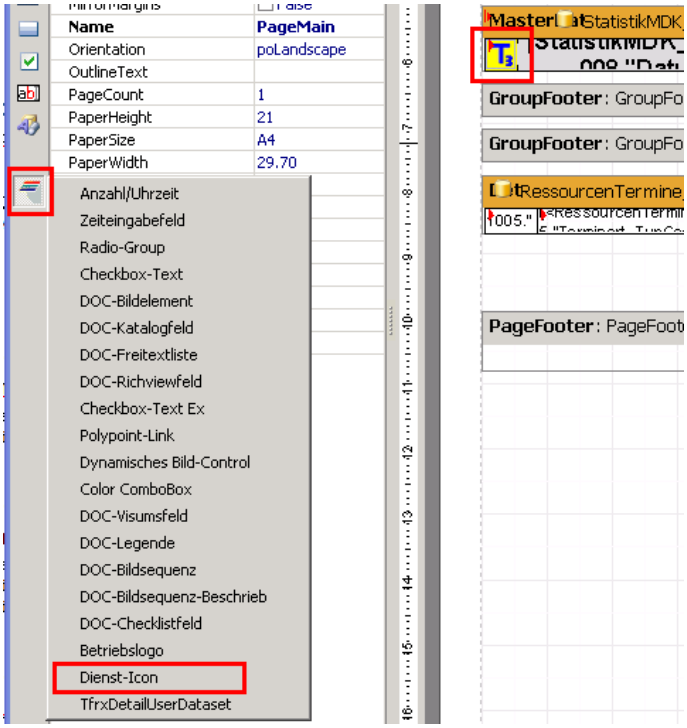
Aus Polypoint_Components „Dienst-Icon“ auswählen und das Ereignis „OnBeforePrint“ aktivieren.

Im SQL-Register folgende Felder einfügen:

```
ma.id as JoinRessourcenID,
tk.knoten_id as JoinKategorieID
```

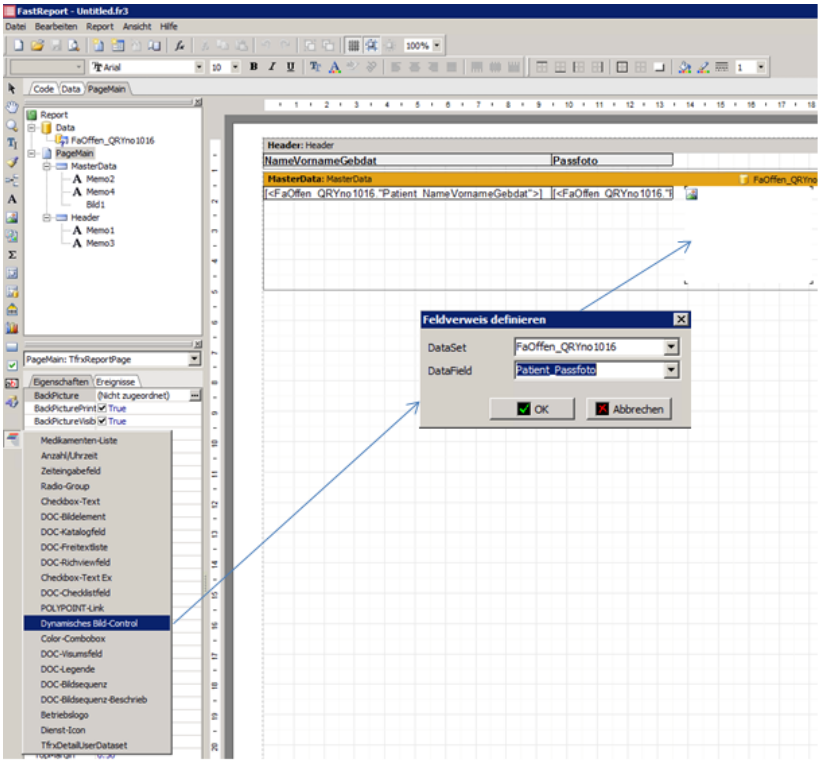
Im Code die Felder entsprechend anpassen.

```
procedure DienstIcon1OnBeforePrint(Sender:
TfrxComponent);
begin
  DienstIcon1.employeeid :=
<StatistikMDK_QRYno3008."JOINRESSOURCENID">
;
  DienstIcon1.categoryid :=
<StatistikMDK_QRYno3008."JOINKATEGORIEID">;
  DienstIcon1.date :=
<StatistikMDK_QRYno3008."Datum">;
end;
```



Feld: Passfoto aus Patientendetail einbinden

Aus Polypoint_Components „Dynamisches Bild-Control“ auswählen und im sich öffnenden Fenster das gewünschte Feld für das Passfoto aus der Abfrage hinterlegen.

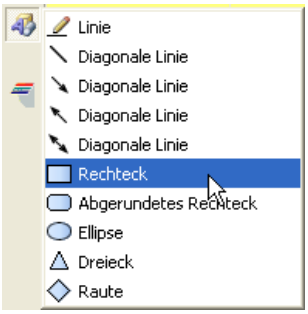


Feld: Terminartfarbe einbinden

Es kann ein Rechteck oder das ganzes Datenfeld eingefärbt werden.

Darauf wird ein Ereignis „OnBeforePrint“ erzeugt und im Code auf das Feld mit dem Farb-Code verwiesen.

```
procedure Shape1OnBeforePrint(Sender: TfrxComponent);
begin
```



```
  TfrxShapeView(Sender).Color := <Ta_QRYno1024."Terminart_Farbe">;
end;
```

Feld: mehrere Feldinhalte verketten

Im Feld Diagnose können mehrere Begriffe ausgewählt werden. Um diese in einem Feld auszugeben, müssen sie mit der Funktion LISTAGG verkettet werden.

```
(select LISTAGG(df.bezeichnung,', ') WITHIN GROUP (ORDER BY df.bezeichnung) from dfitem,
df where dfitem.dfid=df.dfid and dfitem.dftyp = 'Dg' and dfitem.dfkategorie = 'EDg2' and
dfitem.idinanbindung=fa.faid) as Fall_DefDiagnose,

*******************************SQL*******************************************************
Select distinct
to_char(PLANUNG.DATUM,'DY') as TAG1_DATUM,
LISTAGG(MITARBEITER.KUERZEL,', ')WITHIN GROUP (ORDER BY MITARBEITER.KUERZEL) as
TAG1_KUERZEL
from
planung,
mitarbeiter,
dienst
where
PLANUNG.DATUM BETWEEN :dteLeft AND :dteRight
and PLANUNG.KNOTEN_ID in (Select KNOTEN_ID from HIERARCHIE where VATER_ID in ('49575'))
and PLANUNG.MITARBEITER_ID = MITARBEITER.ID
and PLANUNG.PA_CODE = DIENST.PA_CODE
and DIENST.TYP in('P','I')
and PLANUNG.PA_CODE in('1122')
and to_char(PLANUNG.DATUM,'DY')='MO'
group by to_char(PLANUNG.DATUM,'DY')
*****************************************************************************************
```

Formatierungen: Stunden, Minuten

Feldinhalt in Sekunden → Ausgabe: Stunden dezimal

```
[SecondsToHoursDecimals(<StatistikM_QRYno3006."Dienste_Stunden_[111] Dienst A">,2,True)]
```

Für Copy-Paste: Zwischen den <> die Query-Bezeichnung und den Feldnamen eintragen.

```
[SecondsToHoursDecimals(<>,2,True)]
```

Feldinhalt in Sekunden → Ausgabe: Stunden Minuten

```
[SecondsToHoursMinutesSeconds(<StatistikM_QRYno3006."Dienste_Stunden_[112] Dienst
B">,False,True)]
[SecondsToHoursMinutesSeconds(<>,False,True)]
```

Feldinhalt in Sekunden → Ausgabe: Stunden Minuten Sekunden

```
[SecondsToHoursMinutesSeconds(<StatistikM_QRYno3006."Dienste_Stunden_[113] Dienst
C">,True,True)]
[SecondsToHoursMinutesSeconds(<>,True,True)]
```

Formatierungen: Dezimalwerte

Feldinhalt formatieren: (Beispiel: 5 Stellen, 2 Dezimalstellen)

Format('%*.*f', [5, 2, valeur]);

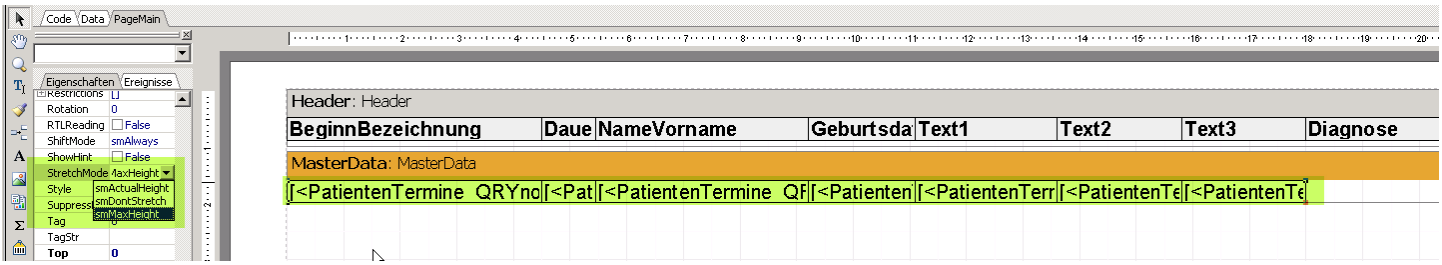
Werte können auch direkt im Feld formatiert werden: z.B

[qryRgLs."TOTAL" #n%2.2n] für 1‘111.11

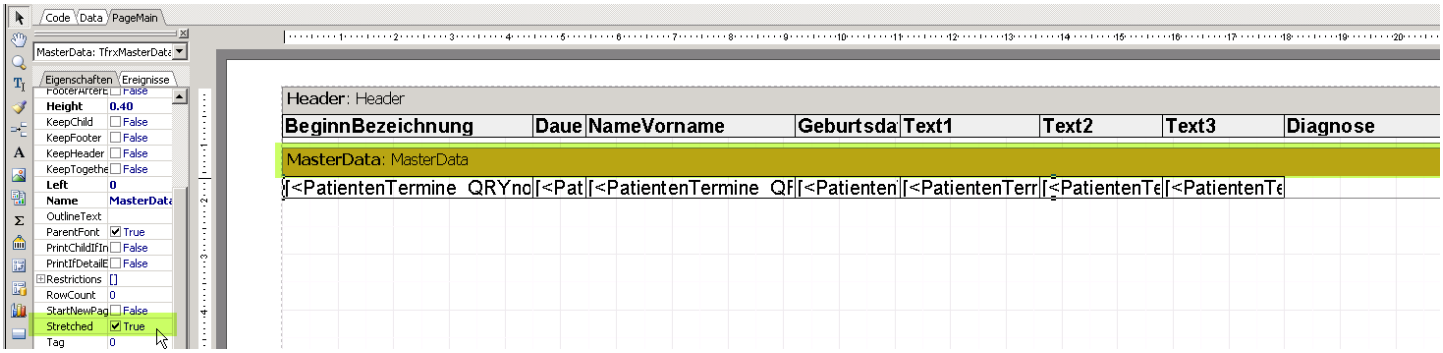
[qryRgLs."TOTAL" #n%2.2f] für 1111.11

Formatierungen: Höhe der Felder optimieren

Alle Felder müssen die Eigenschaft „StretchMode“ auf „smMaxHeight“ gesetzt haben. Dazu alle Felder markieren und Eigenschaft anpassen



Die Eigenschaft „Streched“ des Datencontainers, in diesem Fall das Band „MasterData“ muss auf „True“ gesetzt sein. Dies kann auch via Kontextmenü des Datencontainer und dem Aktivieren der Eigenschaft „Dehnen“ erfolgen.



Formatierungen: Zeilen im Zebralook

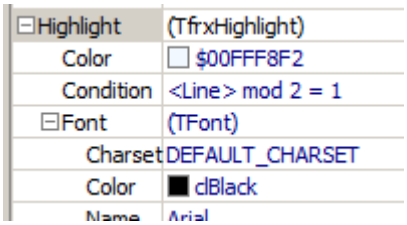
Zebralook einfach

In den Feldeigenschaften:

Highlight – Color: Farbe festlegen

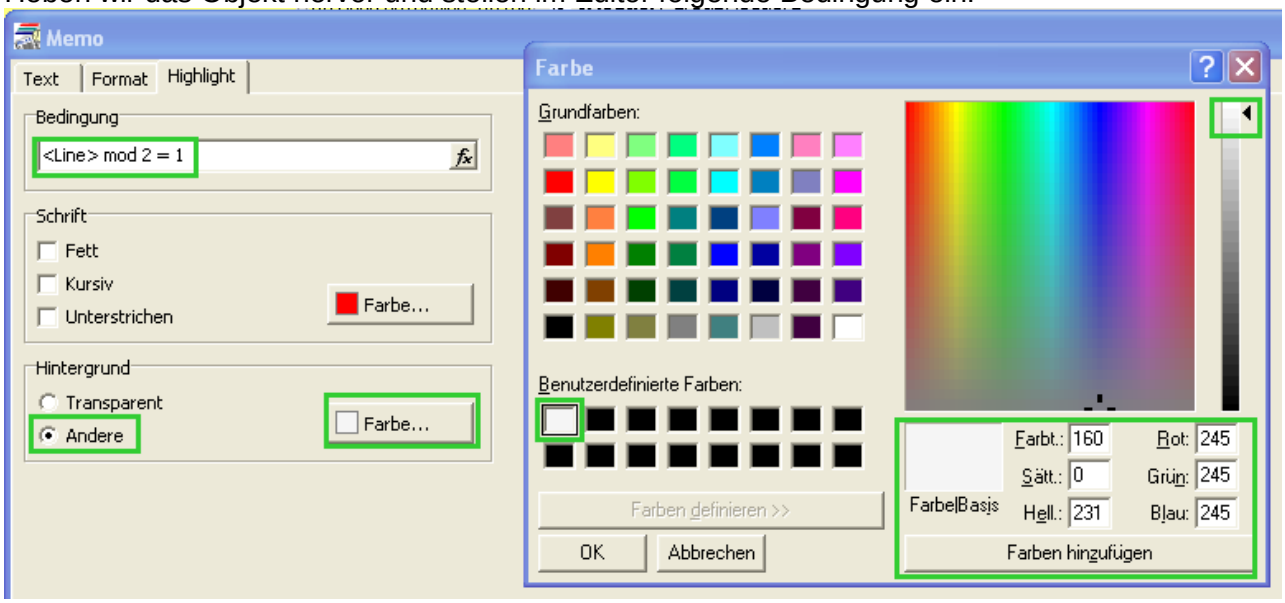
Highlight – Condition: <Line> mod 2 = 1

Highlight – Font – Color: überprüfen



Zebralook kompliziert

Mithilfe der Hervorhebung kann man relativ einfach dem Report ein moderneres Aussehen verpassen, indem man z.B. jede zweite Zeile hervorhebt. Auf das „Masterdata“ legen wir ein Objekt „Text“ und dehnen es auf die komplette Breite des Bandes: Dieses Objekt hat die Rolle einer Unterlage und wird die Farbe ändern, abhängig von der Nummer der Datenzeile.
Heben wir das Objekt hervor und stellen im Editor folgende Bedingung ein:



<Line> mod 2 = 1

Formatierungen: ein Teil des Textes innerhalb eines Text-Objektes formatieren.

In Text-Objekten kann mit HTML-Tags formatiert werden. Vorgängig muss im Kontextmenü des Objektes die Option „HTML Tags erlauben“ aktiviert werden. Folgende Tags werden unterstützt:

 - fetter Text

 - kursiver Text

<u> - unterstrichener Text

<strike> - durchgestrichener Text

<sub> - unterzeiliger Text

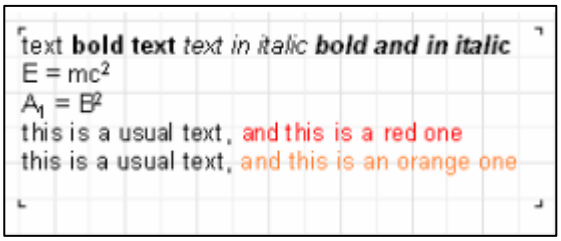
<sup> - Text oberhalb der Zeile

 - Schriftfarbe

<nowrap> - der Text wird nicht wie bei „WordWrap“ getrennt, sondern komplett übertragen

Demonstrieren wir die Anwendungen der Tags an Beispielen.

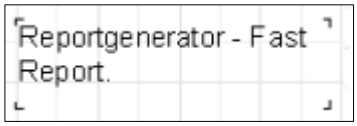
text fetter Text kursiver Text fett und kursiv
E = mc²
A₁ = B²
ein gewöhnlicher Text, und dieser ist rot
ein gewöhnlicher Text, und dieser orange

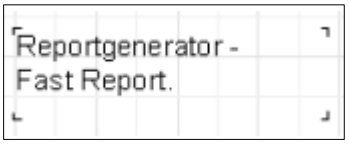


ein Beispiel des Tags <nowrap>.

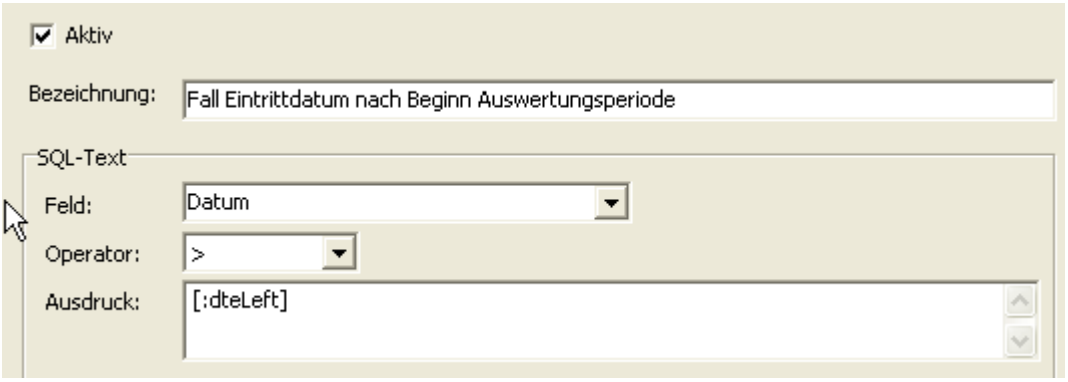
Reportgenerator - Fast Report.

Reportgenerator - <nowrap>Fast Report.</nowrap>

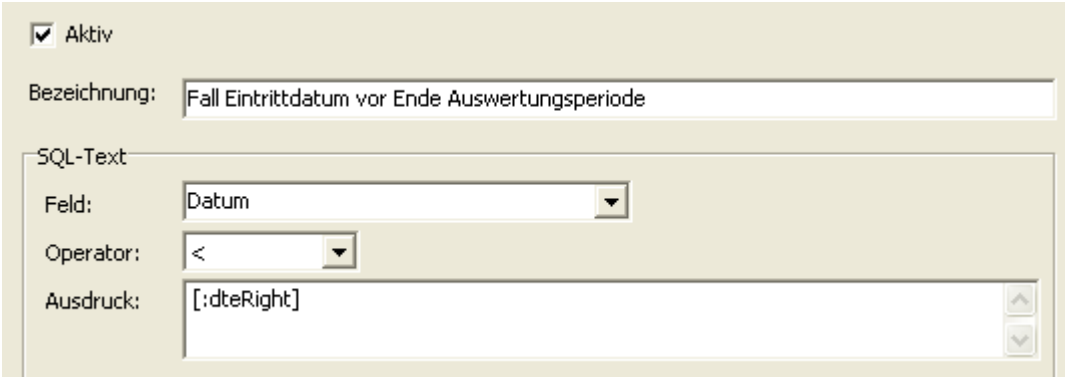




Filter: nur Fälle innerhalb der Auswertungsperiode anzeigen



und



oder

[Fa.Eintritt<=:dteRight AND NVL(Fa.Austritt, :dteLeft)>=:dteLeft]

Filter: nur rein nummerische Personalnummern

```
regexp_instr(rtrim(mitarbeiter.personalnummer),'[A-Z]|[a-z]') = 0
```

Filter: Dienst Beginn und Ende

Dienst Beginn zwischen 12:00 und 23:00 und Dienst Ende zwischen 06:00 und 12:00

```
[
  (TO_CHAR(p.beginn, 'hh24:mi') between '12:00' AND '23:00')
   AND (TO_CHAR(p.ende, 'hh24:mi') between '06:00' AND '12:00')
]
```

Dienst Beginn nach 17:00 und Dienst Ende vor 12:00 oder Dienst Beginn zwischen 00:00 und 06:00 und Dienst Ende vor 12:00

```
[
  (TO_CHAR(p.beginn, 'hh24:mi') > '17:00'
   AND TO_CHAR(p.ende, 'hh24:mi') < '12:00')
or(
  (TO_CHAR(p.beginn, 'hh24:mi') between '00:00' AND '06:00')
   AND TO_CHAR(p.ende, 'hh24:mi') < '12:00'
  )
]
```

Filter: Bei der Auswertung mehrerer Dienste in Spalten, nur MA mit Werten zeigen

Idee: Dienste summieren und Zeile nur ausgeben, wenn die Summe >0 ist. Bsp. Für PA-Code 364,368,369

Vorgehen: NVL plus Funktion des Feldes aus Query

```
[(NVL(GetSumDienstInSekunden(ma.id,s.total,364,GREATEST(:dteLeft,s.gueltig_ab,mea.gueltig
_ab,ks.gueltig_ab),LEAST(:dteRight,s.gueltig_bis,mea.gueltig_bis,ks.gueltig_bis)),0) +
NVL(GetSumDienstInSekunden(ma.id,s.total,368,GREATEST(:dteLeft,s.gueltig_ab,mea.gueltig_a
b,ks.gueltig_ab),LEAST(:dteRight,s.gueltig_bis,mea.gueltig_bis,ks.gueltig_bis)),0) +
NVL(GetSumDienstInSekunden(ma.id,s.total,369,GREATEST(:dteLeft,s.gueltig_ab,mea.gueltig_a
b,ks.gueltig_ab),LEAST(:dteRight,s.gueltig_bis,mea.gueltig_bis,ks.gueltig_bis)),0)) >
'0']
```

Oder mit Query „Personaleinsatzplanung – Statistik pro Mitarbeiter, Datum und Kategorie“

```
[(GetSumServWiNullKatMitF(ma.id,s.total,505,p.datum,p.datum) / 2 >= '0') or
(GetSumServWiNullKatMitF(ma.id,s.total,509,p.datum,p.datum) / 2 >= '0') or
(GetSumServWiNullKatMitF(ma.id,s.total,510,p.datum,p.datum) / 2 >= '0') or
(GetSumServWiNullKatMitF(ma.id,s.total,513,p.datum,p.datum) / 2 >= '0') or
(GetSumServWiNullKatMitF(ma.id,s.total,565,p.datum,p.datum) / 2 >= '0')]
```

Filter: nur Fälle ohne FID

[Fa.FaID is NULL]

Filter: nur der erste Termin des Falles

QueryId: 1009 / Terminplanung_Patienten-Termine

```
[not exists (select 1 from faakt f2, bel b2
where f2.faid=faakt.faid and b2.aktid=f2.aktid and b2.beginn<bel.beginn)]
```

Filter: nach DIS-Stationen in Patienten-Termin Abfragen

Im Tab „SQL“ zum Tab „FROM“ wechseln und die Tabellenbezeichnung **`temp_knoten`** eintragen

Im Tab „WHERE“ den Filter eintragen

```
GetDisStationIDAtTime(SYSDATE, Fa.FaId) in Temp_knoten.knoten_id
```

Filter: Belegungen und leere Betten zum aktuellen Zeitpunkt

QueryId: 1003 / Title: Bettendisposition_Alle Betten

```
[bel.beginn(+) <= :dteLeft + (sysdate - trunc(sysdate)) and
```

```
 bel.ende(+) >= :dteLeft + (sysdate - trunc(sysdate))]
```

Nur die Belegungen:

```
[bel.beginn <= :dteLeft + (sysdate - trunc(sysdate)) and
 bel.ende >= :dteLeft + (sysdate - trunc(sysdate))]
```

Filter: keine gelöschte Termine

```
Akt.Geloescht IN ('F', :AktGeloeschteTermineAnzeigen)
```

oder einfacher:

```
Akt.Geloescht = 'F'
```

Filter: keine gelöschte Termine in Standard-Query aufheben

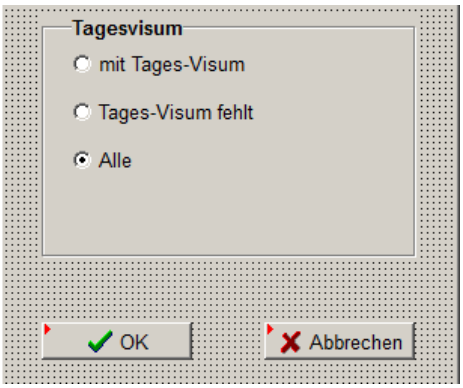
In Standard-Querys werden gelöschte Termine mit folgender Bedingung herausgefiltert:

```
AND NVL(Akt.Geloescht, 'F') IN ('F', :AktGeloeschteTermineAnzeigen)
```

Um die gelöschten Termine trotzdem anzeigen zu können muss du Variable per Script mit `'T'` abgefüllt werden. Dazu wird eine globale Variable i (integer) deklariert und in der globalen Prozedur der folgende Code ausgeführt..

```
var
i     : int;
begin
  for i:= 0 to RessourcenTermine_QRYno1005.Params.Count -1 do
   if RessourcenTermine_QRYno1005.Params.Items[i].name = 'AktGeloeschteTermineAnzeigen'
then
   RessourcenTermine_QRYno1005.Params.Items[i].Value := 'T';
end.
```

Bedingung im Code (if – then – else) für Filter: mit oder ohne Visum




```
procedure BitBtn1OnClick(Sender: TfrxComponent);
begin
  If rb_20.checked then
    begin
     StatistikMDK_QRYno3008.setmacrobyname('mitvisum',' AND p.visum_user_id >0');
    end
  else
  If rb_21.checked then
    begin
     StatistikMDK_QRYno3008.setmacrobyname('ohnevisum',' AND p.visum_user_id is null');
    end
  else
  If rb_22.checked then
   begin
     StatistikMDK_QRYno3008.setmacrobyname('allevisum',' AND 1=1');
   end;
end;
```

Im Where Tab `&mitvisum &ohnevisum &allevisum` einfügen

Bedingung im Code (case): je nach Terminart unterschiedliche Schriftfarbe

```
procedure Memo5OnBeforePrint(Sender: TfrxComponent);
begin
  case <RessourcenTermine_QRYno1005."Terminart_Code"> of
  '1127', '1131', '1133', '1265':
  begin
    TfrxMemoView(Sender).Font.Color := clBlue;
    TfrxMemoView(Sender).Font.Style :=fsBold;
  end;
  '1115':
  begin
    TfrxMemoView(Sender).Font.Color := clGreen;
    TfrxMemoView(Sender).Font.Style :=fsBold;
  end;
  '1280':
  begin
    TfrxMemoView(Sender).Font.Color := clRed;
    TfrxMemoView(Sender).Font.Style :=fsBold;
  end;
'P3840','1110','1112','1114','1116','1117','1118','1119','1120','1121','1129','1149','115
0','1151','1152','1153','1154','1155','1160','1176','1181','1182','1183':
  begin
    TfrxMemoView(Sender).Font.Color := clBlack;
    TfrxMemoView(Sender).Font.Style :=fsBold;
  end;
  else
  begin
    TfrxMemoView(Sender).Font.Color := clBlack;
    TfrxMemoView(Sender).Font.Style := 0;
  end;
  end;
end;
```

Bedingung im Code (stringlist): bestimmter Raum aus String entfernen

Ausgangslage: Im Termin werden mehrere Räume geplant. Auf der Auswertung sollen bestimmte Räume nicht ausgegeben werden.

Mit der Abfrage «Ressourcen-Termine» werden alle im Termin geplanten Räume als String mit Zeilenumbrüchen ausgegeben. Mit dieser Methode werden die in der Stringlist aufgeführten Räume mit einer FOR Schleife nacheinander gesucht, gelöscht und die beiden verbleibenden Teile wieder zusammengesetzt.

```
procedure AuswahlRaumOnBeforePrint(Sender: TfrxComponent);
var
  delCand: TStringList = TStringList.Create;
  raum: TfrxMemoView;

  i, delPos: Integer;
  workingString, part1, part2 : String;

begin
//um weitere Räume auf der Liste zu verbergen --> Stringlist ergänzen
  delCand.add('P2 U1509');  //('132314')
  delCand.add('Aktivität'); //('63347')
  delCand.add('BZ EG 4');  //('63343')

  raum := TfrxMemoView(Sender);

  workingString := <RessourcenTermine_QRYno1005."Termin_Raum_NameAlle">;


  for i:=0 to delCand.Count-1 do
  begin
    delPos := Pos(delCand[i],workingString);

    if delPos > 0 then
    begin
```

```
    part1:= copy(workingString,0,delPos-1);
    part2:= copy(
      workingString,
      delPos + length(delCand[i]) +1,
      length(workingString)
      );
    workingString := part1 + part2;
    end;
  end;
  raum.Text := workingString + <RessourcenTermine_QRYno1005."Termin_Text_3">;
end;
```

Header für weitere Seiten

In Code kopieren

```
procedure CloneComponents(source, target: TfrxComponent; s: String);
var
  i: Integer;
  sourceComp: TfrxComponent;
  targetComp: TfrxComponent;
begin
  target.Assign(source);

  for i := 0 to source.Objects.Count - 1 do
  begin
    sourceComp := TfrxComponent(source.Objects[i]);
    targetComp := nil;

    if sourceComp is TfrxReportPage then
      targetComp := TfrxReportPage.Create(target)
    else
    if sourceComp is TfrxSubreport then
      targetComp := TfrxSubreport.Create(target)
    else
    if sourceComp is TfrxHeader then
      targetComp := TfrxHeader.Create(target)
    else
    if sourceComp is TfrxPageHeader then
      targetComp := TfrxPageHeader.Create(target)
    else
    if sourceComp is TfrxPageFooter then
      targetComp := TfrxPageFooter.Create(target)
    else
    if sourceComp is TfrxSysMemoView then
      targetComp := TfrxSysMemoView.Create(target)
    else
    if sourceComp is TfrxMasterData then
      targetComp := TfrxMasterData.Create(target)
    else
    if sourceComp is TfrxMemoView then
      targetComp := TfrxMemoView.Create(target)
    else
    if sourceComp is TfrxDetailData then
      targetComp := TfrxDetailData.Create(target)
    else
    if sourceComp is TfrxSubDetailData then
      targetComp := TfrxSubDetailData.Create(target)
    else
    if sourceComp is TfrxLogo then
      targetComp := TfrxLogo.Create(target)
    else
    if sourceComp is TfrxLineView then
      targetComp := TfrxLineView.Create(target);

    if targetComp <> nil then
    begin
      targetComp.Name := sourceComp.Name + s;
      CloneComponents(sourceComp,targetComp,s);
```

```
    end;
  end;
end;
```

Im begin – end Teil, der immer am Schluss im Code vorhanden sein muss, die Bezeichnung Source-Page und der Target-Page eintragen werden. Beispiel für drei Seiten:

```
begin
  CloneComponents(PageHeader1, PageHeader2,'p1');
  CloneComponents(PageHeader1, PageHeader3,'p2');
  CloneComponents(PageHeader1, PageHeader4,'p3');
end.
```

Zusätzliche Felder PEP

PEP Personaldaten – Betrieb – Polypoint / RAP

Im SQL-Tab:

Select: `knoten.export_polypoint as Personaldaten_Betrieb_Rap`

From: `knoten`

Where: `knoten.id = ma.ID`

Kostenstelle aus Hierarchie-Gruppe

Geeignet für Abfrage: Personaldaten – Personaldaten gemäss Stichdatum

```
(Select k2.export_code from knoten k2 where k2.id = (select h2.vater_id from hierarchie
h2 where s.kategorie_id=h2.knoten_id))  as Zuordnung_GruppeKST
```

Anzahl Werktage: (mit Q3007)

```
(select sum(decode(datum_typ,'W',1,'H',0.5)) from kalender where kalender_id = 1 and
datum_typ in ('W','H') and datum between '01.01.2010' and '31.12.2010') as xWerktage,
```

Werktage (mit Q3006)

```
decode(k.datum_typ,'W',1,'H',0.5) as Werktage
```

Sollzeit Hundertprozent (mit Q3007)

```
FnVollzeitSoll(ma.id,:dteLeft,:dteright)/3600 as xVollSoll
```

Pikett als Block, Zeitsumme oder Alles

```
GetBlockCount(ma.id,809,To_Date(:dteLeft,'dd.mm.yyyy'),To_Date(:dteRight,'dd.mm.yyyy'),'B
') as Pikett_809_Block,
GetBlockCount(ma.id,809,To_Date(:dteLeft,'dd.mm.yyyy'),To_Date(:dteRight,'dd.mm.yyyy'),'Z
') as Pikett_809_Zeitsumme,
GetBlockCount(ma.id,809,To_Date(:dteLeft,'dd.mm.yyyy'),To_Date(:dteRight,'dd.mm.yyyy'),'A
') as Pikett_809_Alles
```

Guthaben Stichtag Grundguthaben

```
NVL(guthabentagegrund(ma.id, 501, TRUNC(:dteRight,'YEAR')),0)/12* substr(:dteright,4,2)
as StichDat_Grund,
```

Guthaben Stichtag Bezug

```
NVL(GUTHABKOMBITAGEBEZUG(ma.id, 501, :dteRight),0) as StichDat_Bezug
```

Guthaben Bezug vom 1.1. bis Ende der Auswertungsperiode (PA-Code 501)

```
GetSumServWiNullKatMitF(ma.id,s.total,501,GREATEST(trunc(:dteleft,'yyyy'),s.gueltig_ab,me
a.gueltig_ab,ks.gueltig_ab),LEAST(:dteRight,s.gueltig_bis,mea.gueltig_bis,ks.gueltig_bis)
) / 2  as Dienste_Tage_501kum,
```

Planung PA-Code

```
planung.pa_code AS Planung_PACode,
```

Planungs Position

```
DECODE(planung.dienstteil, 'G','Ganz', 'L','Links','R','Rechts','P','Pikett') AS
Planung_Position,
```

Planungs Symbol

```
dienst.bild AS Planung_Symbol
```

Zusätzliche Felder RAP

Patient Name Alphabet

```
Substr(pa.name,1,1) as Patient_AlphabetName
```

Beginn und Ende mit Wochentag und Zeit (inkl.Termine über Mitternacht)

```
Sec2HhMm(Akt.Zeit) || ' - ' || Sec2HhMm(MOD(Akt.Zeit+Akt.Dauer, 24*60*60)) || CASE WHEN
Akt.Zeit+Akt.Dauer>24*60*60 THEN ' (' ||
Substr(To_Char(Akt.datum+TRUNC((Akt.Zeit+Akt.Dauer)/(24*60*60)),'Day'),0,2) || '.)' ELSE
'' END as Termin_VonBisTagAbk
```

Übersetzung Terminart Bezeichnung

Für Abfrage „Patienten – Termine“:

```
GetTaBezTrans(Akt.TaId, Pa.PaId) as Termin_BezTrans
```

Für Abfrage „Ressourcen – Termine“:

```
GetTaBezTrans(Akt.TaId, Bel.PaId) as Termin_BezTrans
```

Für Abfrage „Definitionen - Terminarten“:

1. ID des Sprachcodes (wlta.DFID)mit TAID einer Terminart mit Übersetzung suchen
   ```
   select wlta.WL_TEXT, wlta.DFID, wlta.TAID from wortlisteta wlta where wlta.TAID = 1350
   ```
2. Im folgenden Select wlta.DFID entsprechend anpassen.

```
(select wlta.WL_TEXT  from wortlisteta wlta where wlta.TAID = terminart.TAID and
wlta.DFID = '1012') as BezUebersetzt_deutsch,
(select wlta.WL_TEXT  from wortlisteta wlta where wlta.TAID = terminart.TAID and
wlta.DFID = '1013') as BezUebersetzt_franz,
(select wlta.WL_TEXT  from wortlisteta wlta where wlta.TAID = terminart.TAID and
wlta.DFID = '1014') as BezUebersetzt_ital,
(select wlta.WL_TEXT  from wortlisteta wlta where wlta.TAID = terminart.TAID and
wlta.DFID = '1015') as BezUebersetzt_englisch
```

Übersetzung Raum und Gerät Bezeichnung

Für Abfrage „Patienten – Termine“:

```
GetRsBezTrans(Akt.AktId,'G', -1, Pa.PaId) AS Termin_Geraet_Uebersetzt,
GetRsBezTrans(Akt.AktId,'O', -1, Pa.PaId) AS Termin_Raum_Uebersetzt
```

DIS Urlaub in Query „Alle Betten“

Im SQL-Tab:

Select: **`GetBelInfo(Fa.FaId, 'DU',:dteLeft,:dteRight,'bezeichnung','dd.mm.yyyy') as Aufenthalt_xUrlaub`**

Einzelne Termin Leistungsmarke ausgeben

Termine können mehrere Leistungsmarken haben. Wenn nur eine bestimmte in einem Feld ausgegeben werden soll, muss der entsprechende Code aus dem String der Leistungsmarken Code herausgesucht werden. Bsp. Gesuchte Code ist 100 und gehört zur Leistungsmarke „Bli Bla Blo“

```
case when RTRIM(GetDfByAkt(Akt.AktId, 'Akt', 'Code', 'Mk', 'Mk')) like '%100%' then 'bli
bla blo' end as x100
```

Terminart Farbe als RGB Code ausgeben

In den Standard Querie’s wird die Terminfarbe als einfachen Farbcode ausgegeben. In der Terminart-Definition muss er aber als RGB-Code erfasst werden. Mit den folgenden Zusatzfeldern wird aus dem Farbcode die drei Felder Rot, Grün und Blau errechnet:

```
mod(Terminart.Farbe, 256) as Terminart_Farbe_Rot,
mod(floor(Terminart.Farbe/256), 256) as Terminart_Farbe_Gruen,
mod(floor(Terminart.Farbe/(256*256)), 256) as Terminart_Farbe_Blau
```

Verknüpfung von zwei unterschiedlichen Abfragen:

Einfache Verknüpfung (Fall)

Einträge im Code-Teil:

```
procedure MasterDataOnAfterPrint(Sender: TfrxComponent);
begin

PatientenTermine_QRYno1009.setmacrobyname('faid','fa.faid='+inttostr(<PaGlobal_QRYno1018.
"Interne_FallNummer">));
end;
```

Einträge ganz am Schluss des Code-Teils:

```
begin
  PatientenTermine_QRYno1009.setmacrobyname('faid','1=1');
end.
```

Eintrag im SQL-Teil der entsprechenden Abfrage: Register „Where“

```
&faid
```

Zweifache Verknüpfung (Mitarbeiter; PEP-Planungsdatum/RAP-Termindatum)

Einträge im Code-Teil:

```
procedure MasterDataOnAfterPrint(Sender: TfrxComponent);
begin

RessourcenTermine_QRYno1005.setmacrobyname('rsidplandat','(rs.rsid='+IntToStr(StrToInt(<S
tatistikMDK_QRYno3008."JOINRESSOURCENID">))+') and (akt.datum =
To_Date('''+datetostr(<StatistikMDK_QRYno3008."Datum">)+''','dd.mm.yyyy''))');
end;
```

Einträge ganz am Schluss des Code-Teils:

```
begin
 RessourcenTermine_QRYno1005.setmacrobyname('rsidplandat','1=1');
end.
```

Eintrag im SQL-Teil der entsprechenden Abfrage: Register „Where“

```
&rsidplandat
```

Verknüpfung von zwei nicht-standard Abfragen:

Ausgangslage: Die «QueryPat» erhält aus der tmpintlist und tmpstringlist die paid und faid des aktuellen Patienten. Die «QueryTermin» soll mit der faid verknüpft werden um die entsprechenden Termine zu liefern. Dazu wird in der «QueryTermin» eine Bedingung mit dem Parameter :faid gesetzt.

```
select
faakt.paid,
faakt.faid,
akt.Datum,
akt.bezeichnung
from faakt, akt
where faakt.aktid=akt.aktid
and akt.aktcode='O'
and akt.patientanwesend='T'
and faakt.faid=:faid
and not exists(select 1
               from faakt faakt2, akt akt2
               where faakt2.faid= faakt.faid
               and faakt2.paid=faakt.paid
               and faakt2.aktid=akt2.aktid
               and akt2.datum > akt.datum
               and akt2.zeit > akt.zeit
               and akt2.aktcode='O'
               and akt2.patientanwesend='T')
```

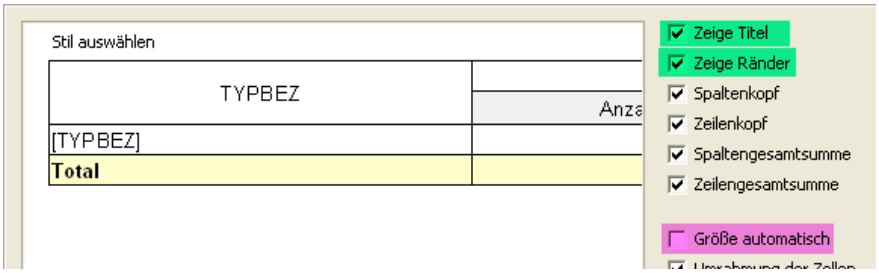
Anschliessend wird im Code des OnBeforePrint-Events des Masterdata die «QueryTermin» geschlossen, der erste Parameter[0] mit der faid gefüllt und die «QueryTermin» wieder geöffnet.

```
procedure MasterData2OnBeforePrint(Sender: TfrxComponent);
begin
  QueryTermin.close;
  QueryTermin.Params[0].value:=<QueryPat."FAID">;
  QueryTermin.Open;
end;
```

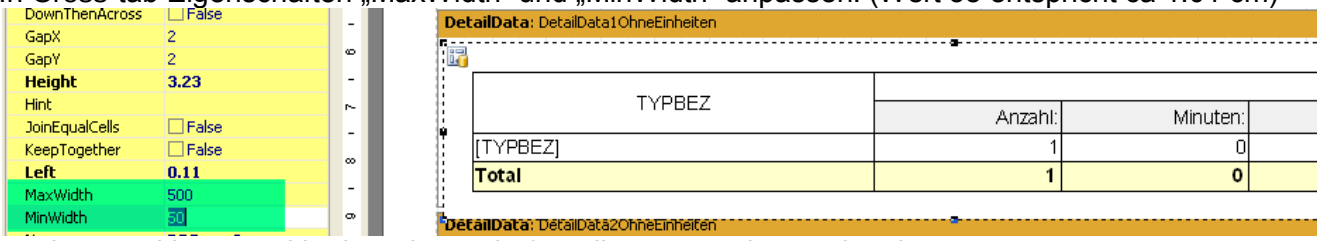
Kreuztabelle formatieren:

Breite der Spalten ändern:

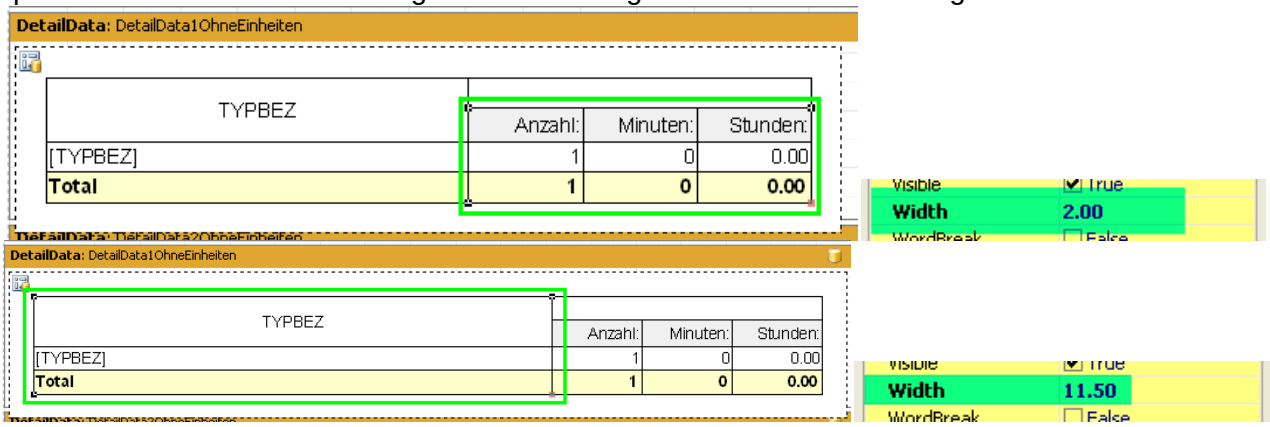
1. Im Cross-tab Editor Optionen „Zeige Titel“ und „Zeige Ränder“ aktivieren. Option „Grösse automatisch“ deaktivieren.



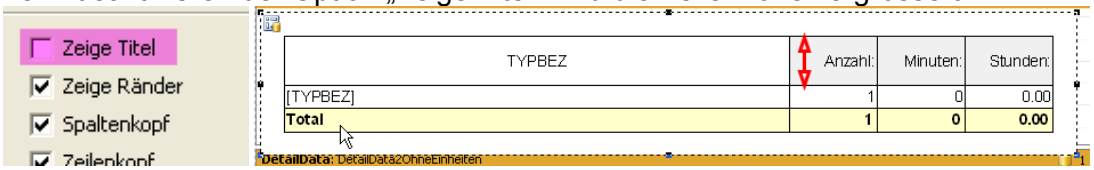
2. In Cross-tab Eigenschaften „MaxWidth“ und „MinWidth“ anpassen. (Wert 38 entspricht ca 1.01 cm)



3. Spalten markieren und in den Eigenschaften die gewünschte Breite eintragen.

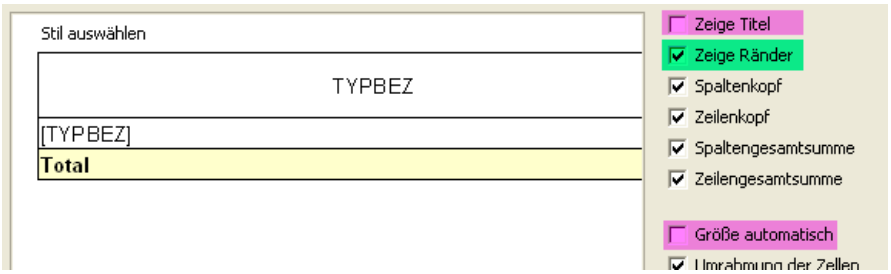


4. Beim deaktivieren der Option „Zeige Titel“ wird die Zeilenhöhe vergrössert.

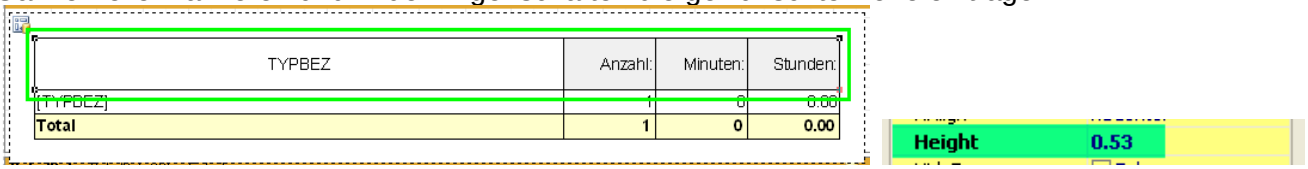


Höhe der Zeilen ändern:

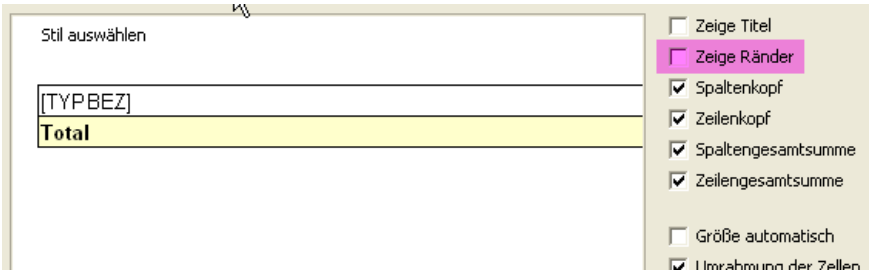
1. Im Cross-tab Editor Option „Zeige Ränder“ aktivieren. Optionen „Zeige Titel“ und „Grösse automatisch“ deaktivieren.



2. Ganze Zeile markieren und in den Eigenschaften die gewünschte Höhe eintragen.



3. Anschliessend kann die Option „Zeige Ränder“ ebenfalls (ohne Risiken und Nebenwirkungen) deaktiviert werden.



Dialogfenster nicht anzeigen bei nur einer Export Seite

Eintrag im Code-Teil

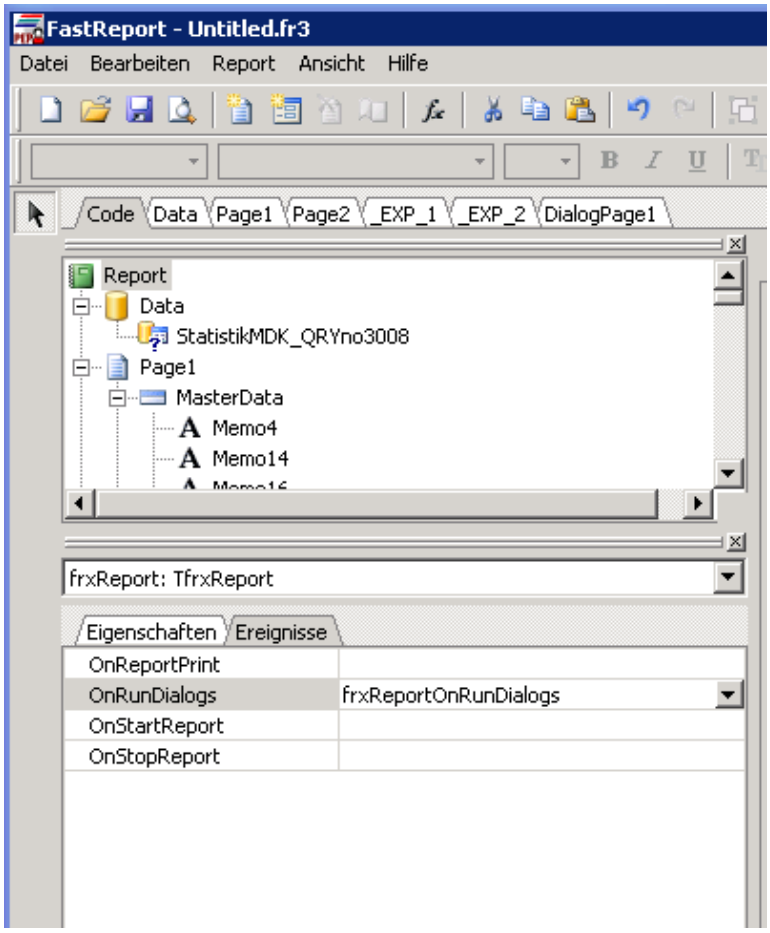
```
procedure frxReportOnRunDialogs(var Result: Boolean);
begin
 if not _EXP_.Visible then
   begin
   if DialogPage1.showmodal = mrCancel then
    Result := False
   else
    Result := True;
   end;
end;
```

Dialogfenster beim Export anzeigen

Eintrag im Code-Teil:

```
procedure frxReportOnRunDialogs(var Result: Boolean);
begin
  if DialogPage1.showmodal = mrCancel then
     Result := False
  else
    Result := True;
end;
```

Zudem in den Eigenschaften des Reports die Prozedur **`frxReportOnRunDialogs`** aktivieren



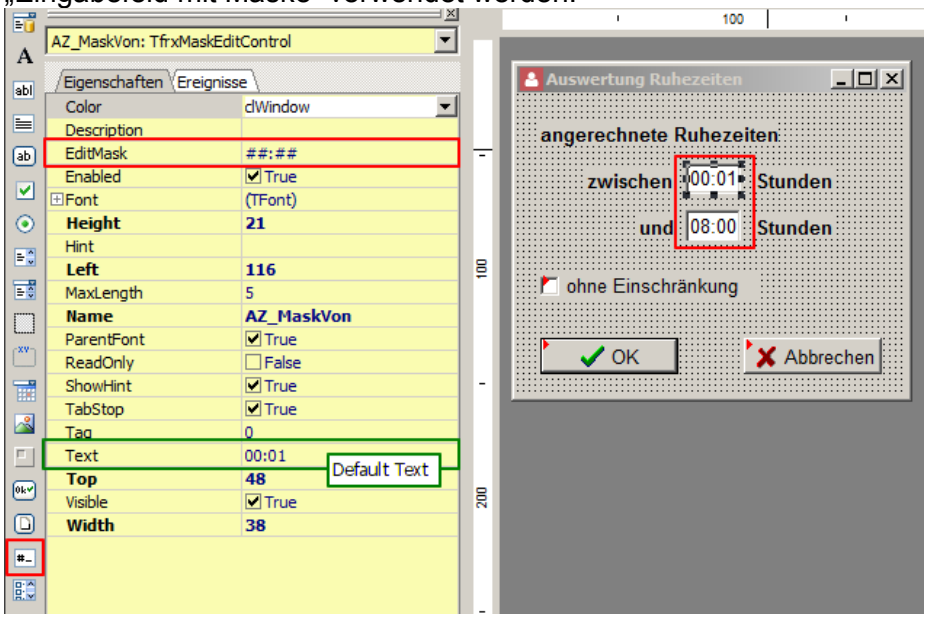
Dialogfenster beim Klick auf „Abbrechen“ schliessen

Ereignis mit folgender Prozedur auf der Schaltfläche „Abbrechen“ erstellen.

```
procedure BitBtn2OnClick(Sender: TfrxComponent);
begin
  engine.StopReport;
end;
```

Dialog mit Eingabefeld mit Maske

Wenn in einem Dialog die Eingaben in einem bestimmten Format erfolgen sollen, kann das Objekt „Eingabefeld mit Maske“ verwendet werden.

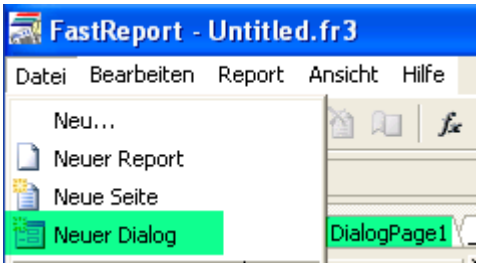


→ Zahlen

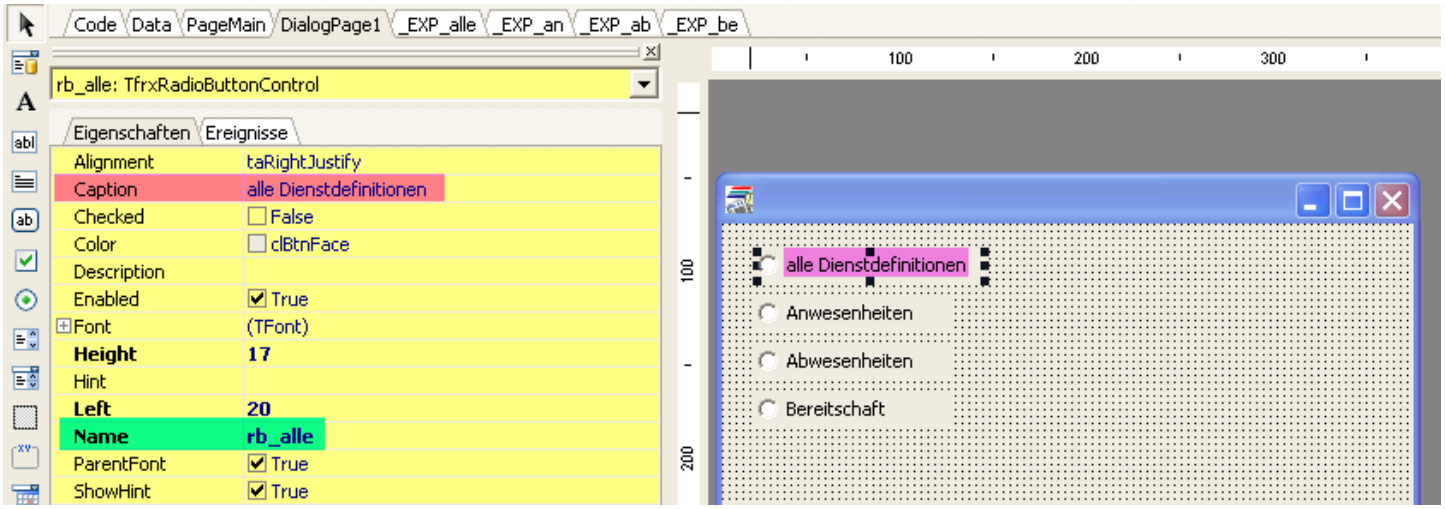
Dialog für mehrere Reportseiten

Variante I

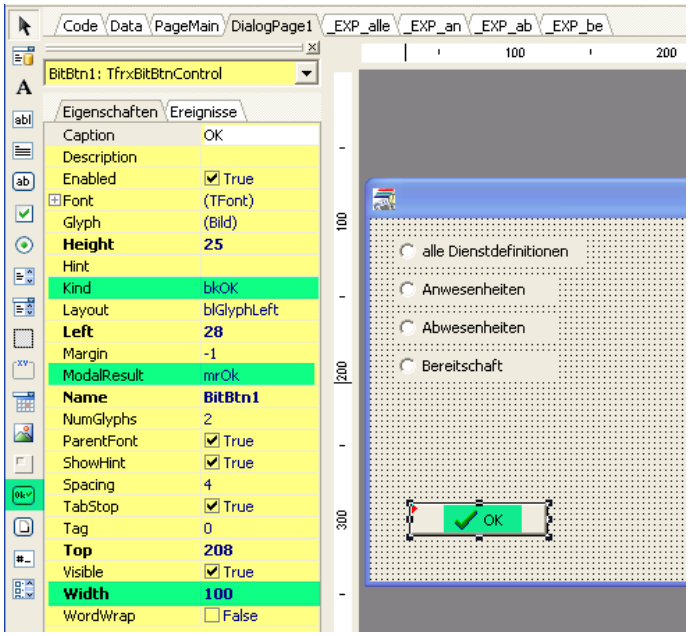
- Neue Dialog-Seite einfügen:

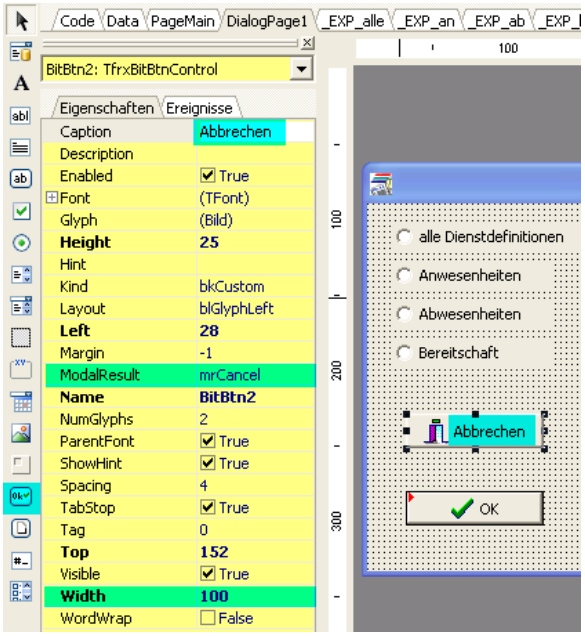


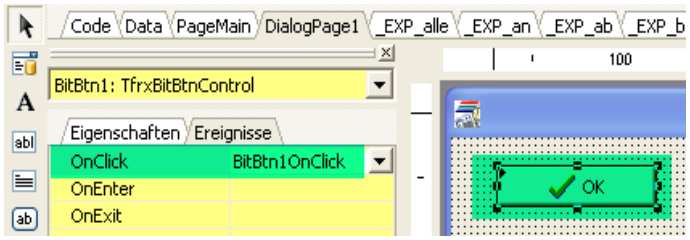
- Radionbuttons einfügen und Caption (Bezeichnung) und Name festlegen



- OK und Abrechen Button einfügen, Ereignis auf OK-Button aktivieren.







- Code entsprechend anpassen resp. ergänzen

```
procedure BitBtn1OnClick(Sender: TfrxComponent);
begin
   if rb_page1.checked then
   begin
```

```
   Page1.Visible := true;
   Page2.Visible := false;
   Page3.Visible := false;
   Page4.Visible := false;
  end
  else
   if rb_page2.checked then
  begin
   Page1.Visible := false;
   Page2.Visible := true;
   Page3.Visible := false;
   Page4.Visible := false;
  end
  else
   if rb_page3.checked then
   begin
   Page1.Visible := false;
   Page2.Visible := false;
   Page3.Visible := true;
   Page4.Visible := false;
  end
 else
   if rb_page4.checked then
   begin
   Page1.Visible := false;
   Page2.Visible := false;
   Page3.Visible := false;
   Page4.Visible := true;
  end;
end;
```

Variante II

```
procedure BitBtn1OnClick(Sender: TfrxComponent);
begin
  Page0.visible := rb_0.checked;
  Page1.visible := rb_1.checked;
  Page2.visible := rb_2.checked;
end;
```