```
fix.with <- function(xx = feb16WithAll.df, out.xls = "MissingWith.xls"){
### Purpose:- Slightly more generic
### ----------------------------------------------------------------------
### Modified from:- fix.septLBAMwith
### ----------------------------------------------------------------------
### Arguments:-
### ----------------------------------------------------------------------
### Author:-   Patrick Connolly, Date:- 17 Feb 2016, 16:02
### ----------------------------------------------------------------------
### Revisions:-

  require(dplyr)

  if(class(xx$Date) != "Date")
    xx <- within(xx, Date <- as.Date(as.character(Date), format = "%d/%m/%Y"))
##   xx <- within(xx, Test <- as.numeric((Efnom)))
##   xx <- within(xx, UC <- unclass(Efnom))
  xx <- within(xx, Efnom <- as.numeric(as.character(Efnom)))
  ##   xx <- xx[!is.na(xx$EfNom),]
  xx <- within(xx, HC <- is.na(Efnom)) # no CO2 (handling control)
  xx <- within(xx, Efnom[is.na(Efnom)] <- 0) # no EF either
  xx <- within(xx, Dead[is.na(Dead)] <- 0) # one empty cell should be zero
  is.egg <- grep("egg", levels(xx$Lifestage), ignore.case = TRUE, value = TRUE)
  xx <- within(xx, IsEgg <- Lifestage%in%is.egg)
  is.scale <- grep("OS", unique(xx$SLS), ignore.case = TRUE, value = TRUE)
  xx <- within(xx, IsScale <- SLS%in%is.scale)
  xx <- xx[!is.na(xx$Total), ] # won't total unless

### Define what is dead
  xx <- within(xx, dead <- Dead) # for eggs will have some eroneously entered as 0
  xx <- within(xx, dead[IsEgg] <- Unhatched[IsEgg]) # overwrites those errors also
  xx <- within(xx, Dead[IsScale] <- Dead[IsScale] + Moribund[IsScale])
  xx$Row <- seq(nrow(xx))

  xxx <- xx %>%
    arrange(Date, SLS, Fruit, Temperature, Duration, Rep, Efpc) %>%
      select(Date, SLS, Fruit, Temperature, Duration, Rep, Efnom, Efpc, HC, dead, Total, Ro
w)
  idcols <- names(xxx %>%
      select(Date, SLS, Fruit, Temperature, Duration, Rep))
  respcols <- names(xxx %>%
                    select(dead, Total, Row))

  xxx <- within(xxx, Ndx <- paste(Date, SLS,Fruit, Temperature, Duration, Rep, sep = "|"))

###  Which are the controls' rows
  xx.hc <- xxx[xxx$HC, ] # i.e. handling controls
  xx.co2c <- xxx[with(xxx, Efpc == 0 & !HC),] # i.e. CO2 controls
  cont.rows <- rbind(xx.hc, xx.co2c)$Row
  treat.rows <- xxx$Row[!xxx$Row %in% cont.rows] # i.e. rows that have treatments applied
  co2cIndx <- with(xx.co2c, Ndx)
  hcIndx <- with(xx.hc, Ndx)
  treatIndx <- unique(xxx$Ndx) # one for every treatment combination
  xx.treat <- xxx[xxx$Row %in% treat.rows,]

### Align controls with the corresponding treated data
  nocont.df <- NULL
  contonly <- NULL
  cont.df <- NULL # collect all control data
  for(i in treatIndx){
##     browser()
    hand.i <- xx.hc[xx.hc$Ndx == i,]
    co2.i <- xx.co2c[xx.co2c$Ndx == i,]
    treat.i <- xx.treat[xx.treat$Ndx == i,]
    cont.i <- NULL
    if(nrow(treat.i) < 1){
      cat(i, "has no treatment data\n")
      contonly <- c(contonly, i)
```

```
    } else {
      ## check if any combinations have not controls
      if(nrow(hand.i) == 0){
        hand.i <- treat.i[1, ]
        hand.i[, respcols] <- NA
        hand.i[, c("Efpc", "HC")] <- c(0, 1)
      }
      cont.i <- rbind(hand.i) # get controls back together
      if(nrow(co2.i) == 0){
        co2.i <- treat.i[1, ]
        co2.i[, respcols] <- NA
        co2.i[, c("Efpc", "HC")] <- c(0, 0)
      }
      cont.i <- rbind(cont.i, co2.i)# get controls back together
      cont.i <- within(cont.i, HC <- as.logical(HC)) # coerced to numeric above
      nocont.i <- rbind(hand.i, co2.i, treat.i)
      nocont.i <- within(nocont.i, HC <- as.logical(HC))
      cont.df <- rbind(cont.df, ditch("Ndx", cont.i))# don't need Ndx
      nocont.df <- rbind(nocont.df, ditch("Ndx", nocont.i))# don't need Ndx
    }
  }
### Get 3 datafranes into one Excel file
  contronly.df <- ditch( "Ndx", xxx[xxx$Ndx %in% contonly,])
  contrmissing.df <- nocont.df[is.na(nocont.df$Row),]

##   require("WriteXLS")
##   WriteXLS(c("cont.df", "contrmissing.df", "contronly.df"), "LBAMmissing.xls",
##            c("AllControls", "NoControls", "ControlsOnly"))

### Use Duration 3 controls when Duration 2 is without
  ## put Ndx back in (slightly different one)
  contrmissing.df <- within(contrmissing.df,
                            Ndx <- paste(Date, HC, SLS, Fruit, Temperature, Duration, Rep,
sep = "|"))
  cont.df <- rbind(cont.df, contronly.df)
  cont.df <- within(cont.df,
                    Ndx <- paste(Date, HC, SLS, Fruit, Temperature, Duration, Rep, sep = "|
"))
##  browser()
  contrmissing.dfB4 <- contrmissing.df
  reused <- 0
  for(k in seq(nrow(contrmissing.df))){
    missing.k <- contrmissing.df[k, ]
    browser()
    if(is.na(missing.k$Total)){ # otherwise nothing needed
      if(missing.k$Duration == 2){
        Ndx.k <- missing.k$Ndx
        Ndx.kFix <- gsub("\\|2\\|", "|3|", Ndx.k)
        ## if(Ndx.k == "2015-05-29|TRUE|LBAM egg|Kiwifruit|5|2|1")
        mort.dat <- c("dead", "Total")
        ##
        reuse.k <- cont.df[cont.df$Ndx == Ndx.kFix, ]
        if(nrow(reuse.k) > 0){
          contrmissing.df[contrmissing.df$Ndx == Ndx.k, mort.dat] <- reuse.k[, mort.dat]
          reused <- reused + 1
          cat("Reused", Ndx.kFix, "\n")
        }
        ##         try(contrmissing.df[contrmissing.df$Ndx == Ndx.k, mort.dat] <-
        ##             cont.df[cont.df$Ndx == Ndx.kFix, mort.dat])
      }
    }
  }

  ## remove contronly.df Rows from repaired control data (useful ones already copied)
  fixed.cont.df <- cont.df[!cont.df$Row %in% contronly.df$Row, ]
  cont.notmissing.df <- contrmissing.df[!with(contrmissing.df, is.na(Total)),]
  use.df <- ditch("Ndx", rbind(fixed.cont.df, cont.notmissing.df, xx.treat)) %>%
                  arrange(Date, SLS, Fruit, Temperature, Duration, Rep, Efnom, Efpc, HC) %>
```

```
%
                    filter(!is.na(Total))
  use.df <- within(use.df,
                   Ndx <- paste(Date, SLS, Fruit, Temperature, Duration, Rep, sep = "|"))
 ##  if(FALSE){

    WriteXLS(c("fixed.cont.df", "contrmissing.dfB4", "contronly.df", "use.df"), out.xls,
             c("AllControls", "NoControls", "ControlsOnly", "ReadyToUse"), BoldHeaderRow =
TRUE,
             FreezeRow = 4, FreezeCol = 3)
 ## }


  ## Check if there's any difference between "Controls"
  if(FALSE){
    test.control2 <- function(dff){
      dff <- dff[dff$Efpc == 0,]  # i.e. controls
      sls <- unique(dff$Ndx)
      cont.out.df <- data.frame(Index = sls)
      cont.out.df <- within(cont.out.df, HC <- CO2 <- Psame <- NA)

      for(sl in sls){
        dfs <- dff[dff$Ndx == sl,]
        cat("\n", sl, ":\n ==================================\n")
        if(nrow(dfs) == 2){
          spec.glm <- glm(cbind(dead, Total - dead) ~ HC, data = dfs,
                          family = binomial)
          hand.mort <- with(dfs[dfs$HC, ], round(100 * dead/Total))
          CO.mort <- with(dfs[!dfs$HC, ], round(100 * dead/Total))
          ##          browser()
          Psl <- anova(spec.glm, test = "Chi")[2, "Pr(>Chi)"]
          cont.out.df <- within(cont.out.df, Psame[Index == sl] <- Psl)
          cont.out.df <- within(cont.out.df, HC[Index == sl] <- hand.mort)
          cont.out.df <- within(cont.out.df, CO2[Index == sl] <- CO.mort)
        }
      }
      cont.out.df
    }

    aa <- test.control2(use.df)
    browser()
    aa[with(aa, HC > CO2),] # shows about 1/3 have HC > CO2
  }
  use.df
}
```