```{r include = FALSE, purl = FALSE, cache = FALSE}

library(knitr)

## set knitr options

opts\_chunk$set(cache = FALSE, tidy.opts = list(width.cutoff = 55), tidy = FALSE, fig.align = "center", fig.width = 5, fig.height = 5, multCode = FALSE, renderTask = TRUE, renderSol = TRUE, renderInfo = TRUE, collapsible = TRUE)

## variables to help save latex answers for appendix

taskCtr <- 0

infoCtr <- 0

solLabels <- list()

infoLabels <- list()

## custom engine for tasks

knit\_engines$set(task = function(options) {

if(options$renderTask) {

## increment counter

assign("taskCtr", taskCtr + 1, pos = 1)

solLabels[[taskCtr]] <- c(taskCtr, NA, NA)

assign("solLabels", solLabels, pos = 1)

## knit child

out <- gsub("``", "```", options$code)

out <- knit\_child(text = out)

## extract header if provided

headerName <- ifelse(length(options$title) > 0, options$title, "Task")

## set up output

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

out <- paste("<div class=\"panel panel-default\"><div class=\"panel-heading\">", headerName, "</div><div class=\"panel-body\">", out, "</div></div>")

} else {

out <- paste0("\\hypertarget{tsk", taskCtr, "}{}\\bblockT[", headerName, "]{\\phantomsection\\label{sol", taskCtr, "}", taskCtr, "}\n", out, "\n\\eblockT\n")

}

return(out)

} else {

return(NULL)

}

})

## custom engine for solutions

knit\_engines$set(solution = function(options) {

if(options$renderSol) {

if(!options$multCode) {

out <- gsub("``", "```", options$code)

out <- knit\_child(text = out)

## extract header if provided

headerName <- ifelse(length(options$title) > 0, options$title, "Solution")

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

out <- paste0("<button id=\"displayText", options$label, "\" onclick=\"javascript:toggle('", options$label, "');\">Show: ", headerName, "</button>\n\n<div id=\"toggleText", options$label, "\" style=\"display: none\"><div class=\"panel panel-default\"><div class=\"panel-heading panel-heading1\"> ", headerName, " </div><div class=\"panel-body\">", out, "</div></div></div>\n")

} else {

solLabels[[taskCtr]][2] <- out

solLabels[[taskCtr]][3] <- headerName

assign("solLabels", solLabels, pos = 1)

<<<<<<< HEAD

out <- paste0("\\hyperlink{sol", taskCtr, "}{\\buttonS{ ", headerName, " on P\\colpageref{tsk", taskCtr, "}}}")

=======

out <- paste0("\\hyperlink{sol", taskCtr, "}{\\buttonS{Show: ", headerName, " on P\\colpageref{tsk", taskCtr, "}}}")

>>>>>>> edac0f594f8a66a049eec537f4636ae64ae7494f

}

return(out)

} else {

## split code

code <- grep("####", options$code)

if(length(code) != 1){

stop(paste("Can't split chunk", options$label, "into two equal parts"))

}

code <- list(options$code[1:(code - 1)], options$code[(code + 1):length(options$code)])

out <- lapply(code, function(x){

gsub("``", "```", x)

})

# tempTidy <- options$tidy

# tempTidyOpts <- options$tidy.opts

out <- lapply(out, function(x) {

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

return(knit\_child(text = x))

} else {

# opts\_chunk$set(tidy = T, tidy.opts = list(width.cutoff = 20))

x <- knit\_child(text = x)

return(x)

}

})

## extract header if provided

headerName <- ifelse(length(options$title) > 0, options$title, "Solution")

## extract headers if provided

if(length(options$titles) == 2){

headerNames <- options$titles

} else {

headerNames <- paste("Option", 1:2)

}

# opts\_chunk$set(tidy = tempTidy, tidy.opts = tempTidyOpts)

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

out <- paste0("<div class=\"tab\"><button class=\"tablinks", options$label, " active\" onclick=\"javascript:openCode(event, 'option1", options$label, "', '", options$label, "');\">", headerNames[1], "</button><button class=\"tablinks", options$label, "\" onclick=\"javascript:openCode(event, 'option2", options$label, "', '", options$label, "');\">", headerNames[2], "</button></div><div id=\"option1", options$label, "\" class=\"tabcontent", options$label, "\">", out[[1]], "</div><div id=\"option2", options$label, "\" class=\"tabcontent", options$label, "\">", out[[2]], "\n</div><script> javascript:hide('option2", options$label, "') </script>")

out <- paste0("<button id=\"displayText", options$label, "\" onclick=\"javascript:toggle('", options$label, "');\">Show: ", headerName, "</button>\n\n<div id=\"toggleText", options$label, "\" style=\"display: none\"><div class=\"panel panel-default\"><div class=\"panel-heading panel-heading1\"> ", headerName, " </div><div class=\"panel-body\">", out, "</div></div></div>\n")

return(out)

} else {

out <- paste0("\\bmp\n\\bblockST{", headerNames[1], "}\n", out[[1]], "\n\\eblockST\n\\emp\n\\hspace{0.01\\textwidth}\n\\bmp\\bblockST{", headerNames[2], "}\n", out[[2]], "\n\\eblockST\n\\emp\n")

solLabels[[taskCtr]][2] <- out

solLabels[[taskCtr]][3] <- headerName

assign("solLabels", solLabels, pos = 1)

out <- paste0("\\hyperlink{sol", taskCtr, "}{\\buttonS{Show: ", headerName, " on P\\colpageref{tsk", taskCtr, "}}}")

}

return(out)

}

} else {

return(NULL)

}

})

## custom engine for information boxes

knit\_engines$set(info = function(options) {

if(options$renderInfo) {

## increment counter

assign("infoCtr", infoCtr + 1, pos = 1)

infoLabels[[infoCtr]] <- c(infoCtr, NA, NA)

assign("infoLabels", infoLabels, pos = 1)

## parse text

out <- gsub("``", "```", options$code)

out <- knit\_child(text = out)

## extract header if provided

headerName <- ifelse(length(options$title) > 0, options$title, "Info")

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

if(options$collapsible) {

## set up environment with collapsible button

out <- paste0("<infobutton id=\"displayText", options$label, "\" onclick=\"javascript:toggle('", options$label, "');\">Show: ", headerName, "</infobutton>\n\n<div id=\"toggleText", options$label, "\" style=\"display: none\"><div class=\"panel panel-default\"><div class=\"panel-body\">", out, "</div></div></div>\n")

} else {

## set up standard boxed environment

out <- paste("<div class=\"panel panel-default\"><div class=\"panel-heading\">", headerName, "</div><div class=\"panel-body\">", out, "</div></div>")

}

} else {

if(options$collapsible) {

## add info to headers

infoLabels[[infoCtr]][2] <- out

infoLabels[[infoCtr]][3] <- headerName

assign("infoLabels", infoLabels, pos = 1)

out <- paste0("\\hypertarget{infoRet", infoCtr, "}{}\n\\hyperlink{info", infoCtr, "}{\\buttonS{Show: ", headerName, " on P\\colpageref{info", infoCtr, "}}}")

out <- paste0(out, "\n\\phantomsection\\label{infoRet", infoCtr, "}")

} else {

out <- paste0("\\bblockT[", headerName, "]\n", out, "\n\\eblockT\n")

}

}

} else {

out <- NULL

}

return(out)

})

## custom engine for tabbed box

knit\_engines$set(multCode = function(options) {

## split code

code <- grep("####", options$code)

if(length(code) != 1){

stop(paste("Can't split chunk", options$label, "into two equal parts"))

}

code <- list(options$code[1:(code - 1)], options$code[(code + 1):length(options$code)])

out <- lapply(code, function(x){

gsub("``", "```", x)

})

# tempTidy <- options$tidy

# tempTidyOpts <- options$tidy.opts

out <- lapply(out, function(x) {

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

return(knit\_child(text = x))

} else {

# opts\_chunk$set(tidy = T, tidy.opts = list(width.cutoff = 20))

x <- knit\_child(text = x)

return(x)

}

})

## extract headers if provided

if(length(options$titles) == 2){

headerNames <- options$titles

} else {

headerNames <- paste("Option", 1:2)

}

# opts\_chunk$set(tidy = tempTidy, tidy.opts = tempTidyOpts)

if(opts\_knit$get("rmarkdown.pandoc.to") != "latex") {

out <- paste0("<div class=\"tab\"><button class=\"tablinks", options$label, " active\" onclick=\"javascript:openCode(event, 'option1", options$label, "', '", options$label, "');\">", headerNames[1], "</button><button class=\"tablinks", options$label, "\" onclick=\"javascript:openCode(event, 'option2", options$label, "', '", options$label, "');\">", headerNames[2], "</button></div><div id=\"option1", options$label, "\" class=\"tabcontent", options$label, "\">", out[[1]], "</div><div id=\"option2", options$label, "\" class=\"tabcontent", options$label, "\">", out[[2]], "</div><script> javascript:hide('option2", options$label, "') </script>")

} else {

out <- paste0("\\bmp\n\\bblockST{", headerNames[1], "}\n", out[[1]], "\n\\eblockST\n\\emp\n\\hspace{0.01\\textwidth}\n\\bmp\\bblockST{", headerNames[2], "}\n", out[[2]], "\n\\eblockST\n\\emp\n")

}

return(out)

})

```