Wtorek 17:00

# Code review

# Potencjalne błędy przy zmianie zamówienia

Na pewno będzie nam zależało, żeby nasz skrypt reagował na każdą zmianę wartości – np. zaznaczenie którejś opcji. Naturalnie, możesz pomyśleć, że kiedy zaznaczymy jakąś opcję, to jej opcji powinna zostać dodana lub odjęta od ceny produktu. Takie podejście jest jednak ryzykowne, ponieważ jest podatne na błędy.

Nie będziemy teraz wchodzić w bardzo techniczne szczegóły potencjalnych problemów z takim podejściem, ale są sytuacje, w których pola formularza mogą zmieniać wartość, a nasz skrypt nie będzie o tym powiadomiony. Przykładem może być wtyczka do przeglądarki, która zapamiętuje stan formularza.

Użytkownik może wejść na stronę, zaznaczyć wszystkie składniki, a następnie odświeżyć stronę – jego wtyczka wtedy mogłaby zaznaczyć wszystkie składniki bez informowania o tym naszego skryptu. Jest to sytuacja, której być może nie da się uniknąć, ale jeszcze gorzej, jeśli użytkownik wtedy zacznie odznaczać składniki, a cena produktu zacznie spadać – może nawet poniżej zera!

# \_ vs \_\_

'.product\_\_total-price .price',

# Jak działa ta funkcja

utils.serializeFormToObject = function(*form*){

  let output = {};

  if (typeof *form* == 'object' && *form*.nodeName == 'FORM') {

    for (let field of *form*.elements) {

      if (field.name && !field.disabled && field.type != 'file' && field.type != 'reset' && field.type != 'submit' && field.type != 'button') {

        if (field.type == 'select-multiple') {

          for (let option of field.options) {

            if(option.selected) {

              utils.createPropIfUndefined(output, field.name);

              output[field.name].push(option.value);

            }

          }

        } else if ((field.type != 'checkbox' && field.type != 'radio') || field.checked) {

          utils.createPropIfUndefined(output, field.name);

          output[field.name].push(field.value);

        } else if(!output[field.name]) output[field.name] = [];

      }

    }

  }

  return output;

};

# Dlaczego console.log('aaabbbb'); się nie pokazuje?

          if (formData[singlParamId].includes(singlParamOption)) {

            if (option.hasOwnProperty('default')) {

              console.log('pppppp');

            }

            else {

              console.log('aaabbbb');

              price += option.price;

            }

          }

          else {

            if (option.hasOwnProperty('default') && option.default === true) {

              price -= option.price;

            }

          }

# Github problem z rebase

if (formData[singlParamId] && formData[singlParamId].includes(singlParamOption)) {

            for (const inclOptions of formData[singlParamId]) {

              const allExtraImgIncl = thisProduct.element.querySelector(`${thisProduct.imageWrapper} img.${singlParamId}-${inclOptions}`);

              if (allExtraImgIncl) {

                allExtraImgIncl.classList.add(classNames.menuProduct.imageVisible);

              }

            }

            if (!option.hasOwnProperty('default') || option.default === false) {

              price += option.price;

podświetlona linia występowała w gałęzi bocznej, a nie w main, github nie widział problemu

# Czemu nie można użyć zapisu: thisWidget.elem = *elem*; w konstruktorze?

  class AmountWidget {

    constructor(*elem*) {

      const thisWidget = *this*;

      thisWidget.getElements(*elem*);

      // console.log('thisWidget)', thisWidget);

      // console.log('constructor', elem);

    }

    getElements(*elem*) {

      const thisWidget = *this*;

      thisWidget.elem = *elem*;

      thisWidget.input = thisWidget.elem.querySelector('input');

      thisWidget.btnLess = thisWidget.elem.querySelector('a[href\*="less"].btn-quantity ');

      thisWidget.btnMore = thisWidget.elem.querySelector('a[href\*="#more"].btn-quantity ');

      console.log('aaaaa', thisWidget.btnLess);

    }

  }

Dlaczego !thisWidget.value === newValue zamiast thisWidget.value !== newValue && !isNaN(newValue)  nie zadziała?

    setValue(*value*) {

      const thisWidget = *this*;

      const newValue = parseInt(*value*);

      const maxValue = settings.amountWidget.defaultMax;

      const minValue = settings.amountWidget.defaultMin;

      thisWidget.value = settings.amountWidget.defaultValue;

      if (thisWidget.value !== newValue && !isNaN(newValue) && newValue >= minValue && newValue <= maxValue) {

        thisWidget.value = newValue;

        thisWidget.announce();

      }

      thisWidget.input.value = thisWidget.value;

    }