* **Redux-saga**
  + Xử lý side effect
  + Xử lý data phức tạp
  + Sử dụng Generator function

Viết code async như sync

* + Thường dùng cho ứng dụng lớn, phức tạp
* Các method
  + Fork(): sử dụng cơ chế non - blocking call trên function
  + Call(): Gọi function. Nếu nó return về một promise, tạm dừng saga cho đến khi promise được giải quyết
  + Take(): tạm dừng cho đến khi nhận được action
  + Put(): Dùng để dispatch một action
  + takeEvery(): Theo dõi một action nào đó thay đổi thì gọi một saga nào đó
  + akeLastest() : Có nghĩa là nếu chúng ta thực hiện một loạt các actions, nó sẽ chỉ thực thi và trả lại kết quả của của actions cuối cùng
  + yield(): Có nghĩa là chạy tuần tự khi nào trả ra kết quả mới thực thi tiếp
  + Select(): Chạy một selector function để lấy data từ state

Fork: thực hiện một hoạt động non-blocking trên function được truyền cho nó.

Take: tạm dừng cho đến khi nhận được action

Race: chạy nhiều effect đồng thời, sau đó hủy tất cả nếu một trong số đó kết thúc.

Call: gọi function. Nếu nó return về một promise, tạm dừng saga cho đến khi promise được giải quyết.

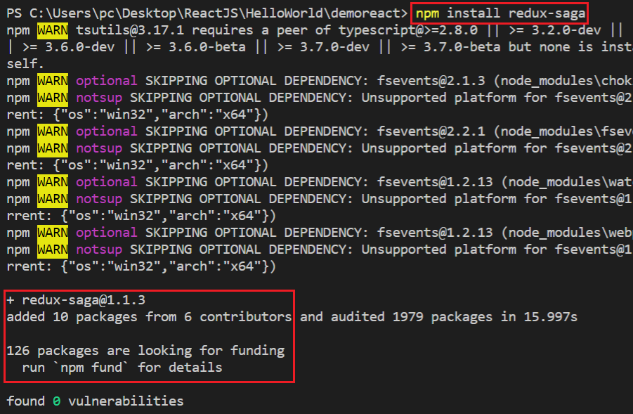
Put: dispatch một action.

Select: chạy một selector function để lấy data từ state.

takeLatest: có nghĩa là nếu chúng ta thực hiện một loạt các actions, nó sẽ chỉ thực thi và trả lại kết quả của của actions cuối cùng.

takeEvery: thực thi và trả lại kết quả của mọi actions được gọi.

* Install



*npm install redux-saga*

tách side effect khỏi action bằng cách đặt chúng vào các saga

* **Reference**

<https://redux-saga.js.org/>